MATEMATIKOS IR INFORMATIKOS INSTITUTAS

Gintautas DZEMYDA
Olga KURASOVA
Julius ŽILINSKAS

DAUGIAMAČIŲ DUOMENŲ VIZUALIZAVIMO METODAI

Vadovėlis informatikos krypties doktorantams ir magistrantams

MOKSLO AIDAI

Vilnius 2008

UDK 004.932(075.8)
Dz-08

Lietuvos Respublikos švietimo ir mokslo ministerijos Aukštųjų mokyklų bendrųjų vadovėlių leidybos komisijos rekomenduota 2008 03 28 Nr. 08-390

Recenzentai:
prof. habil. dr. Rimantas Šeinauskas (Kauno technologijos universitetas)
prof. habil. dr. Jonas Mockus (Matematikos ir informatikos institutas)

Redaktorė Zita Manstavičienė

ISBN 978-9986-680-42-0

Turinys

Pratarmė

Technikoje, medicinoje, ekonomikoje, ekologijoje ir daugelyje kitų sričių nuolatos susiduriama su daugiamačiais duomenimis. Faktiškai nėra žmonių veiklos srities, kur nebūtų kaupiami ir analizuojami tokie duomenys. O vystantis technologijoms, tobulėjant kompiuteriams ir programinei įrangai kaupiamų duomenų apimtys ypač sparčiai didėja. Auga ir poreikiai bei nauda, gaunama padarius teisingas išvadas.

Klasikinis daugiamačių duomenų pavyzdys – R. Fišerio dar 1936 metais paskelbta keturmačių duomenų lentelė. Šie duomenys vis dar dažnai naudojami įvairių duomenų analizės metodų eksperimentiniuose tyrimuose. Buvo išmatuoti trijų veislių 150-ies irisų (vilkdalgių) žiedų vainiklapių pločiai, vainiklapių ilgiai, taurėlapių pločiai ir taurėlapių ilgiai. Taigi kiekvienas žiedas apibūdinamas keturiais skaičiais (požymiais, parametrais). Geometriškai – tai 150 taškų, išsibarsčiusių keturmatėje erdvėje. Kitas pavyzdys – krūties vėžio duomenys. Moterims, kurioms diagnozuotas piktybinis ar nepiktybinis krūties navikas, buvo matuoti devyni fiziologiniai parametrai. Gauti 683 taškai, išsibarstę devynmatėje erdvėje.

Yra daugybė daugiamačių duomenų analizės metodų. Tai įvairūs klasifikavimo, klasterizavimo, statistinės analizės ir kiti metodai. Jais galima nustatyti stebimų duomenų artimumą, sudaryti taisykles, pagal kurias tokio tipo duomenys būtų rūšiuojami, vertinti atskirų parametrų įtaką daromam sprendimui. Tačiau dažnai žmogus nori neformaliai pažinti turimus duomenis, kad tikrai pasitikėtų daromo sprendimo kokybe ir patikimumu. Vienas iš būdų – savo akimis pasižiūrėti į turimų duomenų visumą. Ar įmanoma tai padaryti, pamatyti, kaip išsidėstę erdvėje keturmačiai ar devynmačiai taškai? Visais atvejais tai skamba viliojančiai, nes tai tarsi pasižvalgymas po daugiamatę erdvę.

Didelio matmenų skaičiaus duomenų vizualizavimas suteikia galimybę tyrinėtojui pačiam stebėti tų duomenų grupavimosi tendencijas, įvertinti atskirų daugiamačių taškų tarpusavio artimumą, racionaliai priimti sprendimus. Ne veltui yra labai daug bandymų „nugalėti“ ir pažinti daugiamatiškumą. Siekiant gauti kuo daugiau naujų žinių apie analizuojamus duomenis, bandoma net sujungti kelis skirtingais principais grindžiamus vizualizavimo metodus.

Šis daugiamačių duomenų vizualizavimo metodų vadovėlis skirtas aukštųjų mokyklų informatikos ir informatikos inžinerijos krypties studentams bakalaurams, magistrantams ir doktorantams. Vadovėlyje pateiktus vizualios

analizės metodus gali taikyti įvairių sričių specialistai. Nors pasaulyje mokomosios ir mokslinės literatūros apie duomenų gavybos ir analizės, kartu ir daugiamačių duomenų vizualizavimo metodus yra daug, tačiau lietuvių kalba tai pirmasis vadovėlis iš šios tematikos. Dėl šios srities literatūros lietuvių kalba stokos kai kurių terminų vertimas iš anglų kalbos į lietuvių kalbą nėra nusistovėjęs, todėl vadovėlyje dėl aiškumo prie lietuviškų terminų pateikiami ir angliškieji.

Visi trys autoriai yra parašę daug mokslinių darbų daugiamačių duomenų vizualizavimo srityje. Jie šia tema yra paskelbę daugiau kaip 60 mokslinių publikacijų Lietuvos bei tarptautiniuose moksliniuose žurnaluose, knygose ir enciklopedijose, yra skaitę kviestinius pranešimus tarptautinėse konferencijose ir doktorantų vasaros mokyklose. Pirmas plataus masto su daugiamačių duomenų vizualiosios analizės taikymais susijęs ir juos integruojantis su kitais duomenų analizės metodais darbas lietuvių kalba yra G. Dzemydos, P. Gudyno, V. Šaltenio ir V. Tiešio 2001 metais išleista monografija *Lietuvos pedagogai ir moksleiviai: analizė ir prognozė*.

Vadovėlio autoriai daug metų skaito paskaitas Vilniaus pedagoginio universiteto ir Vilniaus Gedimino technikos universiteto studentams: bakalaurams ir magistrantams, yra paruošę studijų modulius Matematikos ir informatikos instituto doktorantams, parengė keturis vadovėlius, taip pat du studijų kursus internete. Sukaupta patirtis turėjo tiesioginės įtakos šio vadovėlio turiniui ir medžiagos pateikimo formai.

Vadovėlis yra skirtas studijuoti nuosekliai, ypač kai naudojamas kaip pagrindinis daugiamačių duomenų vizualizavimo moduliuose. Tačiau jis yra parašytas taip, kad skyriai būtų kaip galima labiau nepriklausomi ir galėtų būti studijuojami atskirai, papildant gretimų modulių suteiktas žinias arba studijuojant savarankiškai. Tie, kas nori susipažinti su daugiamačių duomenų vizualizavimo strategijomis, turėtų skaityti 2 skyrių. Tačiau jau susipažinę ir norintys gilintis į daugiamačių skalių arba dirbtinių neuroninių tinklų metodus daugiamačiams duomenims vizualizuoti gali iš karto atversti 3 arba 4 skyrių. 5 skyriuje aprašyti vizualiosios analizės taikymai įvairiose srityse: socialiniuose moksluose, medicinoje, farmakologijoje ir kitur. Norintys greitai susipažinti su vizualiosios analizės teikiamais privalumais gali pirmiausia peržvelgti taikymų skyrių, o tada gilintis į metodus, kurie jiems pasirodė labiausiai tinkami konkrečiai sričiai. Platus literatūros sąrašas, kurio didžiumą sudaro moksliniai rezultatai daugiamačių duomenų analizės ir vizualizavimo srityje, leis giliau pažinti vadovėlyje pateiktas idėjas ir metodus.

Žymėjimai

c_{kl}	parametrų x_k ir x_l kovariacijos koeficientas
$C = \{c_{kl}, k, l = 1, \ldots, n\}$	kovariacinė matrica
d	vaizdo erdvės (projekcinės erdvės, į kurią atvaizduojamas n-matis vektorius) matmenų skaičius, $d < n$
$d(X_k, X_l)$	atstumas tarp n-mačių vektorių X_k ir X_l
$d(Y_k, Y_l)$	atstumas tarp d-mačių vektorių Y_k ir Y_l, $d < n$
$d_q(Y_k, Y_l)$	Minkovskio atstumas tarp vektorių Y_k ir Y_l
δ_{ij}	i-ojo ir j-ojo objektų skirtingumas
E_{DS}	daugiamačių skalių (DS) paklaida
E^*	minimali santykinė daugiamačių skalių paklaida
E_S	Sammono projekcijos paklaida
$E_{SOM(q)}$, $E_{SOM(t)}$	saviorganizuojančio neuroninio tinklo (SOM) kvantavimo ir topografinė paklaidos
$E(Y)$	santykinė daugiamačių skalių paklaida
E_k	matricos k-asis tikrinis vektorius
η	gradientinio optimizavimo žingsnio ilgį reguliuojantis parametras
k_x, k_y	SOM tinklo eilučių ir stulpelių skaičiai
L	paslėptųjų neuronų sluoksnių skaičius
λ_k	matricos k-ąjį tikrinį vektorių atitinkanti tikrinė reikšmė
n	vektoriaus komponenčių skaičius; erdvės, kuriai priklauso vektorius X_i, matmenų skaičius; objektą apibūdinančių parametrų skaičius
n_l	paslėptųjų neuronų skaičius l-ajame neuroninio tinklo sluoksnyje
m	analizuojamų objektų (vektorių) skaičius

M_{ij}	SOM tinklo neuronai
M_c	SOM tinklo neuronas nugalėtojas
r_{kl}	parametrų x_k ir x_l koreliacijos koeficientas
$R = \{r_{kl}, k, l = 1, \ldots, n\}$	koreliacinė matrica
R^n	n-matė erdvė
$S(Y)$	mažiausiųjų kvadratų įtempimo funkcija *Stress*
S^*	mažiausiųjų kvadratų įtempimo funkcijos minimumas
T_i	norimų neuroninio tinklo išėjimo reikšmių vektorius įėjimo vektoriui X_i
w_{jk}	jungties iš *k*-ojo įėjimo į *j*-ąjį neuroną svoris neuroniniame tinkle; svoris daugiamatėse skalėse
x_{ij}	duomenų vektoriaus X_i *j*-osios komponentės reikšmė; objektą X_i apibūdinančio *j*-ojo parametro reikšmė
x_j	objektą apibūdinantis *j*-asis parametras
$X = \{X_1, X_2, \ldots, X_m\}$	analizuojamų duomenų matrica, kurios *i*-oji eilutė yra *n*-matis vektorius X_i
$X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$	*i*-asis duomenų vektorius, $X_i \in R^n$
X_i^T	transponuotasis vektorius X_i
$Y = (Y_1, Y_2, \ldots, Y_m)$	vaizdo taškų rinkinys
Y^*	vaizdo taškų rinkinys esant minimaliai įtempimo funkcijai
$Y_i = (y_{i1}, y_{i2}, \ldots, y_{id})$	vaizdo taško koordinatės, vektoriaus X_i transformacija į mažesnio skaičiaus matmenų (vaizdo) erdvę R^d, $d < n$
y_j	vaizdo taško *j*-oji koordinatė; neuroninio tinklo *j*-ojo išėjimo reikšmė
$\| X_k - X_l \|$	Euklido atstumas tarp vektorių X_k ir X_l

1. Įvadas

Kasdien susiduriame su duomenimis, informacija ir žiniomis. Šios sąvokos viena su kita yra glaudžiai susijusios, tačiau iš esmės – labai skirtingos. *Duomenys* – tai objektyviai egzistuojantys faktai, vaizdai arba garsai, kurie gali būti naudingi tam tikram uždaviniui spręsti. *Informacija* – tai duomenys, kurių forma ir turinys yra pateikti tinkamu naudoti sprendimų priėmimo procese būdu. Duomenys virsta informacija, kai jiems suteikiamas kontekstas ir jie susiejami su tam tikra problema ar sprendimu. *Žinios* yra gebėjimas spręsti problemas, atnaujinti arba sukurti naujas vertes remiantis ankstesne patirtimi, įgūdžiais ar išmokimu. Tai žmogaus proto abstrakcija apie duomenis, jų prasmę, naudą ir sąryšius. Turimos žinios gali virsti informacija, kuri gali būti panaudota naujoms žinioms įgyti.

Šiuolaikiškomis technologijomis galima gauti ir saugoti didelius duomenų kiekius ir srautus. Tačiau tebelieka esminė problema – kaip tuos duomenis suvokti ir interpretuoti, o taip pat priimti geriausius sprendimus žinių, gautų iš šių duomenų, pagrindu. Dažnai labai svarbu gauti kuo daugiau naudingų žinių, suprasti duomenis, atskirti svarbią informaciją nuo menkavertės. Suvokti duomenis yra nelengvas uždavinys, ypač kai jie yra *daugiamačiai*, t. y. kai nurodo sudėtingą objektą ar reiškinį, apibūdinamą daugeliu parametrų (požymių, savybių, rodiklių, ypatybių), kurie gali būti ne tik skaitiniai, bet loginiai, tekstiniai ir kt. Dažnai iškyla būtinybė nustatyti ir giliau pažinti tokių duomenų struktūrą: susidariusias grupes (klasterius, *cluster*), itin išsiskiriančius objektus (taškus atsiskyrėlius, *outliers*), objektų tarpusavio panašumą ar skirtingumą ir pan.

Apibrėžkime pagrindines sąvokas ir žymėjimus, vartojamus šiame vadovėlyje. Nagrinėsime daugiamačius duomenis, jų vizualiosios analizės problemas ir metodus. Čia susiduriame su dviem pagrindinėmis sąvokomis. Tai *objektas* ir *parametras*. Sąvoka *objektas* gali apimti įvairius dalykus: žmones, įrenginius, gamybos produktus, augalus, gamtos reiškinius ir kt. Objektai (*objects*), sudarantys konkrečią analizuojamų objektų aibę, yra apibūdinami bendrais parametrais (*features*, *parameters, attributes*), dar dažnai vadinamais požymiais, savybėmis, rodikliais, ypatybėmis. Objektų tokioje aibėje skaičius m yra baigtinis, tačiau bendru atveju gali būti ir labai didelis. Tam tikras visų parametrų reikšmių rinkinys nusako vieną konkretų analizuojamos aibės objektą $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $i \in \{1, \ldots, m\}$, čia n yra parametrų skaičius, i yra objekto eilės numeris. Parametrų skaičius n dar yra vadinamas duomenų

matmenų skaičiumi. Kai objektą $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$ apibūdina daugiau nei vienas parametras, duomenys (objektai) yra *daugiamačiai*.

Objektai $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$ dažnai vadinami vektoriais ar taškais, parametrai $x_1, x_2, \ldots, x_n$ – komponentėmis. Šiame vadovėlyje šios sąvokos taip pat bus naudojamos, kai susidursime su matematinėmis uždavinių formuluotėmis ir sprendimu.

Vadovėlyje nagrinėsime pakankamai bendrą duomenų analizės atvejį, kai parametrai įgyja tam tikras skaitines reikšmes. Todėl analizuojamų duomenų aibė yra matrica

$$X = \{X_1, X_2, \ldots, X_m\} = \{x_{ij},\ i = 1, \ldots, m,\ \ j = 1, \ldots, n\},$$

kurios *i*-oji eilutė yra vektorius $X_i \in R^n$, čia $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $i \in \{1, \ldots, m\}$, m – analizuojamų objektų (vektorių) skaičius.

Būna atvejų, kai atskirą objektą nusakančio parametrų skaitinių reikšmių rinkinio neturime, jo negalima gauti, t. y. negalime suformuoti skaitinių duomenų matricos *X*, tačiau galime skaitiškai įvertinti artimumus (*proximity*) tarp objektų porų – panašumus (*similarity*) ar skirtingumus (*dissimilarity*). Šiuo atveju duomenų matmenų skaičius *n* gali būti ir nežinomas. Taip dažnai būna, pavyzdžiui, psichologiniuose tyrimuose. Tuo atveju analizė atliekama naudojantis artimumų (pavyzdžiui, skirtingumų) matrica

$$\Delta = \{\delta_{ij}, i, j = 1, \ldots, m\},$$

čia δ_{ij} yra objektų X_i ir X_j artimumo įvertis. Artimumų matricą galima gauti ir iš matricos *X*, pritaikius kokią nors artimumo metriką. Daugiamačių duomenų vizualizavimas artimumų matricos pagrindu taip pat nagrinėjamas šiame vadovėlyje.

1.1. Vizualizavimo tikslai

Kuo didesnės apimties duomenys (didesnis objektų arba matmenų skaičius), tuo sunkiau iš duomenų lentelės suvokti objektų visumos ypatybes. Tokių duomenų analizė reikalauja intensyvių tyrimų. Tai gana sudėtingi uždaviniai. Spręsdamas juos žmogus gali įsigilinti į duomenis ir daryti išvadas. Tam tikslui yra naudojami įvairūs duomenų analizės metodai: klasifikavimo, klasterizavimo, vizualizavimo ir kt. Šiuo metu duomenų analizė yra labai aktuali įvairiose veiklos srityse: versle, ekonomikoje, medicinoje, sociologijoje ir kt. Labai svarbu duomenis pateikti žmogui suprantama forma, padedančia

geriau juos suvokti: nustatyti struktūrą, tarpusavio ryšius, susidariusias grupes, prognozuojamus įverčius ir pan. Vienas iš galimų būdų yra vizualizavimas. *Vizualizavimas* – tai grafinis informacijos pateikimas. Pagrindinė vizualizavimo idėja – duomenis pateikti tokia forma, kuri leistų tyrėjui juos suprasti, daryti išvadas ir tiesioginę įtaką tolesniam jų srautui. Vizualizavimas leidžia geriau suvokti sudėtingas duomenų aibes, gali padėti nustatyti dominančius jos poaibius. Vizualią informaciją žmogus pajėgus suvokti daug greičiau negu tekstinę, ji palengvina naujų žinių atradimą.

Duomenų vizualizavimo sąvoka yra gana plati. Turint duomenis lentelės pavidalu, galima nubraižyti histogramas, prognozės grafikus ir kitas diagramas, iš kurių išvadas apie duomenis galima padaryti daug lengviau nei iš skaitinių duomenų lentelės. Tai paprasčiausi būdai duomenims vizualizuoti.

Šiame vadovėlyje nagrinėjama klasė metodų, kurie vadinami *daugiamačių duomenų vizualizavimo metodais*. Jie padeda nustatyti ar įvertinti daugiamačių duomenų struktūrą: susidariusias grupes (klasterius), itin išsiskiriančius objektus (taškus atsiskyrėlius), panašumus tarp analizuojamų objektų ar jų grupių ir pan.

Per pastaruosius 40 metų daugiamačių duomenų vizualizavimo metodai intensyviai plėtojami siekiant didinti duomenų analizės efektyvumą, suprantamiau pateikti ir objektyviau įvertinti duomenų gavybos bei analizės rezultatus. Šie metodai plėtojami dviem pagrindinėmis kryptimis. Tiesioginio vizualizavimo metodais kiekvienas daugiamačio objekto parametras yra pateikiamas tam tikra vizualia forma. Projekcijos, dar vadinamieji matmenų skaičiaus mažinimo metodai (*dimension reduction techniques*), transformuoja analizuojamų duomenų aibę $X = \{X_1, X_2, \ldots, X_m\}$ iš n-matės erdvės R^n į mažesnio matmenų skaičiaus vaizdo erdvę R^d ($d < n$), kur duomenų aibės transformacijos

$$Y = \{Y_1, Y_2, \ldots, Y_m\} = \{y_{ij},\ i = 1, \ldots, m,\ \ j = 1, \ldots, d\}$$

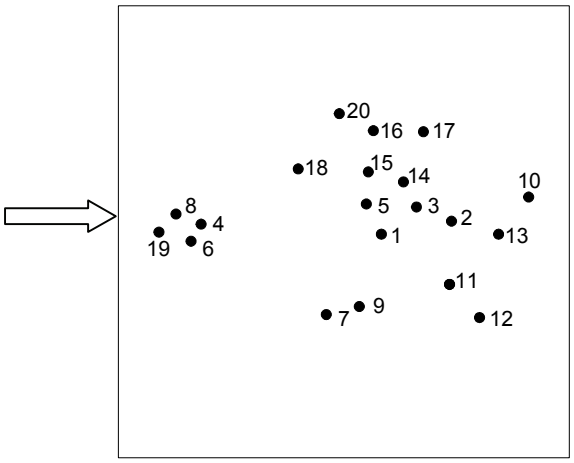
taškų išsidėstymą galima stebėti vizualiai. Pastarojo tipo metodams, įskaitant ir su jais susijusius dirbtinius neuroninius tinklus, skiriama didžioji šio vadovėlio dalis. Atskiras skyrius skirtas metodams, kuriuose vietoje aibės $X = \{X_1, X_2, \ldots, X_m\}$ gali būti panaudota ir matrica $\Delta = \{\delta_{ij},\ i, j = 1, \ldots, m\}$ – objektų artimumų matrica. Transformavus projekcijos metodais daugiamačius duomenis į dvimatę ar trimatę vaizdo erdvę ir juos vizualizavus, daug paprasčiau suvokti duomenų struktūrą ir sąryšius tarp jų. Tačiau duomenis transformuojant į mažesnio skaičiaus matmenų erdvę, neišvengiami duomenų

iškraipymai ir paklaidos. Kaip jas minimizuoti – aktuali problema, plačiai nagrinėjama vadovėlyje.

Yra išskiriami du pagrindiniai duomenų vizualizavimo tikslai:

1. Tiriamoji analizė (*explorative analysis*), kai vizualizavus duomenis iškeliamos hipotezės.
2. Patvirtinančioji analizė (*confirmative analysis*), kai vizualizavus duomenis patvirtinamos jau esančios hipotezės.

	x_1	x_2	x_3	x_4	x_5	x_6
X_1	0,57	0,11	0,47	0,54	0,84	0,53
X_2	0,65	0,14	0,53	0,55	0,89	0,54
X_3	0,61	0,13	0,49	0,52	0,92	0,57
X_4	0,44	0,16	0,45	0,42	0,99	0,45
X_5	0,55	0,11	0,53	0,52	0,92	0,55
X_6	0,44	0,16	0,43	0,41	0,98	0,43
X_7	0,49	0,12	0,56	0,46	0,82	0,48
X_8	0,41	0,17	0,42	0,43	0,98	0,42
X_9	0,54	0,12	0,59	0,46	0,85	0,48
X_{10}	0,69	0,16	0,63	0,56	0,91	0,49
X_{11}	0,66	0,16	0,51	0,48	0,87	0,55
X_{12}	0,70	0,13	0,52	0,48	0,81	0,52
X_{13}	0,65	0,14	0,63	0,52	0,89	0,49
X_{14}	0,59	0,11	0,51	0,57	0,89	0,52
X_{15}	0,53	0,11	0,52	0,57	0,89	0,50
X_{16}	0,52	0,13	0,51	0,61	0,92	0,51
X_{17}	0,58	0,12	0,55	0,61	0,95	0,52
X_{18}	0,45	0,10	0,52	0,53	0,92	0,51
X_{19}	0,42	0,17	0,41	0,42	0,98	0,40
X_{20}	0,53	0,11	0,42	0,57	0,91	0,57



1.1 pav. Daugiamačių duomenų vizualizavimo pavyzdys

Klasterių paieškos daugiamačiuose duomenyse jų vizualizavimo būdu pavyzdys pateiktas 1.1 paveiksle. Klasterizavimas (*clustering*) – tai toks analizuojamų objektų suskirstymas į skirtingas grupes, dar vadinamus klasterius (*clusters*), kad grupės objektai būtų panašūs tarpusavyje, o objektai iš skirtingų grupių būtų nepanašūs.

Lentelėje nurodyti daugiamačiai duomenys – vektoriai $X_1, X_2, \ldots, X_m$, čia $m = 20,$ kurių kiekvieną apibūdina šeši parametrai, taigi duomenų matmenų skaičius $n = 6.$ Šie duomenys iš daugiamatės erdvės R^6 atvaizduojami plokštumoje R^2 naudojantis vadinamuoju daugiamačių skalių metodu, kuris detaliai aprašytas 3 skyriuje. Taigi vaizdo erdvė yra dvimatė. 1.1 paveiksle daugiamačiai duomenų vektoriai $X_1, X_2, \ldots, X_{20}$ vaizduojami taškais, skaičiai atitinka vektorių indeksus (numerius). Matome, kad vektorius X_4, X_6, X_8 ir X_{19} atitinkantys taškai sudaro atskirą klasterį, t. y. jie visi nutolę nuo kitų taškų, o patys yra arti vienas kito. Šį klasterį galima aiškiai matyti plokštumoje, tačiau sunku nustatyti iš duomenų lentelės be tam tikros papildomos analizės. Kita vertus, toks vizualus duomenų pateikimas leidžia tyrinėtojui „pajusti“ daugiamačių vektorių tarpusavio atstumus, o tai palengvina duomenų visumos pažinimą.

1.2. Artimumo metrikų apibrėžimai

Daugiamačius duomenis vizualizuojant ir klasterizuojant, būtina apibrėžti duomenų artimumo matą. Dažnai artimumo matas yra Euklido metrika, kuri priklauso Minkovskio metrikos grupei.

Minkovskio metrikoje, esant fiksuotam skaičiui *q*, atstumas tarp dviejų objektų $X_k = (x_{k1}, x_{k2}, \ldots, x_{kn})$ ir $X_l = (x_{l1}, x_{l2}, \ldots, x_{ln})$ apskaičiuojamas pagal šią formulę:

$$d_q(X_k, X_l) = \left\{ \sum_{j=1}^{n} | x_{kj} - x_{lj} |^q \right\}^{\frac{1}{q}}.$$

Šiai metrikų šeimai priklauso:

- *Miesto kvartalo* (*city-block, Manhattan*) metrika ($q = 1$), kurioje atstumas

$$d_1(X_k, X_l) = \sum_{j=1}^{n} | x_{kj} - x_{lj} |;$$

- *Euklido* metrika ($q = 2$), kurioje atstumas

$$d_2(X_k, X_l) = \sqrt{\sum_{j=1}^{n} | x_{kj} - x_{lj} |^2};$$

- *Čebyšovo* metrika ($q = \infty$), kurioje atstumas

$$d_\infty(X_k, X_l) = \max_j |x_{kj} - x_{lj}|.$$

Atstumas tarp dviejų objektų X_k ir X_l Minkovskio metrikoje tenkina tokias sąlygas:

- $d(X_k, X_l)$ yra neneigiamas realusis skaičius, $d(X_k, X_k) = 0$;
- $d(X_k, X_l) = d(X_l, X_k)$, t. y. atstumas nuo objekto X_k iki objekto X_l yra lygus atstumui nuo objekto X_l iki objekto X_k;
- $d(X_k, X_l) \le d(X_k, X_j) + d(X_j, X_l)$, t. y. atstumas tarp bet kurių dviejų objektų X_k ir X_l negali būti didesnis nei suma atstumų tarp objektų X_k, X_j ir X_l, X_j (trikampio nelygybė).

Tuo atveju, kai atskirą objektą nusakančio parametrų skaitinių reikšmių rinkinio negalima gauti, tenka ekspertiškai ar kokiu nors kitu būdu skaitiškai įvertinti artimumus (*proximity*) tarp objektų porų – panašumus (*similarity*) ar skirtingumus (*dissimilarity*). Šiuo atveju aukščiau minėta trikampio nelygybė gali negalioti.

1.3. Duomenų normavimas

Tegul analizuojamų duomenų aibė $X = \{X_1, X_2, \ldots, X_m\}$ yra sudaryta iš *n*-mačių vektorių $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $i = 1, \ldots, m$, t. y. matricos *X i*-oji eilutė yra vektorius X_i.

Paprastai konkretaus analizuojamo daugiamačių duomenų rinkinio $X = \{X_1, X_2, \ldots, X_m\}$ parametrų $x_1, x_2, \ldots, x_n$ reikšmės kinta skirtinguose intervaluose arba jos išreikštos skirtingais matavimo vienetais (pavyzdžiui, kilogramai, metrai, laipsniai). Todėl prieš analizuojant duomenis būtina suvienodinti šių reikšmių mastelius. Tai galima atlikti keliais būdais. Pateikiame du iš jų:

1. Parametrų reikšmės pakeičiamos tokiomis, kurių vidurkiai lygūs nuliui, o dispersijos lygios vienetui. Tai atliekama apskaičiuojant kiekvieno parametro x_j vidurkį

$$\overline{x}_j = \frac{1}{m} \sum_{i=1}^{m} x_{ij}$$

ir dispersiją

$$\sigma_j^2 = \frac{1}{m-1}\sum_{i=1}^{m}(x_{ij} - \bar{x}_j)^2 .$$

Kiekviena to parametro reikšmė x_{ij} transformuojama pagal formulę

$$x_{ij} = \frac{(x_{ij} - \bar{x}_j)}{\sqrt{\sigma_j^2}} .$$

2. Parametrų reikšmės pakeičiamos tokiomis, kurių minimalios reikšmės lygios nuliui, o maksimalios reikšmės lygios vienetui. Čia apskaičiuojamos kiekvieno parametro x_j minimali $x_{j\min}$ ir maksimali $x_{j\max}$ reikšmės, kiekviena to parametro reikšmė x_{ij} transformuojama pagal formulę

$$x_{ij} = \frac{(x_{ij} - x_{j\min})}{(x_{j\max} - x_{j\min})} .$$

1.4. Testiniai duomenys

Šiame vadovėlyje įvairių metodų veikimui iliustruoti naudojami keli pagrindiniai duomenų rinkiniai, kurie aprašyti šiame skyrelyje. Visus juos galima rasti duomenų bazėje „UCI Repository of Machine Learning Databases“ (http://archive.ics.uci.edu/ml/). Tačiau vien šiais duomenimis vadovėlyje neapsiribojama – gausu kitų testinių ar praktinės kilmės duomenų.

Fišerio irisų duomenys [51], kurie kartais vadinami tiesiog irisais arba irisų duomenimis, yra vieni iš klasikinių testinių duomenų, naudojamų daugiamačių duomenų analizei. Išmatuoti keturi 150-ies irisų (vilkdalgių) žiedų parametrai:

- taurėlapių ilgiai (*sepal length*);
- taurėlapių pločiai (*sepal width*);
- vainiklapių ilgiai (*petal length*);
- vainiklapių pločiai (*petal width*).

Matuotos trijų veislių gėlės: *Iris Setosa* (I klasė), *Iris Versicolor* (II klasė) ir *Iris Virginica* (III klasė). Sudaryti 4-mačiai ($n = 4$) vektoriai $X_1, X_2, \ldots, X_{150}$, čia $X_i = (x_{i1}, x_{i2}, x_{i3}, x_{i4})$, $i = 1, \ldots, 150$.

Įvairiais metodais nustatyta, kad I klasės irisai skiriasi nuo kitų dviejų (II ir III) klasių, o pastarųjų – labiau giminingi.

Automobilių, pagamintų 1970–1982 metais JAV, Europoje ir Japonijoje, duomenys (398 automobiliai). Trijų regionų automobilius apibūdina devyni parametrai:

- degalų sąnaudos (*mpg*);
- cilindrų skaičius (*cylinders*);
- variklio darbo tūris (*displacement*);
- arklio jėgų kiekis (*horsepower*);
- svoris (*weight*);
- greitis (*accelerator*);
- modelio pagaminimo metai (*model years*);
- kilmės regionas (*origin*);
- modelis (*name*).

Paskutiniai du parametrai nėra skaitiniai, todėl paprastai vizualizavimo procese tiesiogiai nėra naudojami – jie nusako objektų priklausomybę skirtingoms klasėms, pavyzdžiui, skirtingi kilmės regionai ar modeliai. Dažniausiai analizuojami 7-mačiai ($n = 7$) vektoriai $X_1, X_2, \ldots, X_{398}$, čia $X_i = (x_{i1}, x_{i2}, \ldots, x_{i7})$, $i = 1, \ldots, 398$.

Krūties vėžio duomenys (683 atvejai) surinkti Viskonsino universiteto (JAV) ligoninėje gydytojo dr. W. H. Wolbergo [105]. Du naviko tipus – nepiktybinį (*benign*) ir piktybinį (*malignant*) – apibūdina devyni skaitiniai parametrai:

- klampumas (*clump thickness*);
- ląstelės dydžio vienodumas (*uniformity of cell size*);
- ląstelės formos vienodumas (*uniformity of cell shape*);
- kraštinis sulipimas (*marginal adhesion*);
- vienasluoksnio epitelio ląstelės dydis (*single epithelial cell size*);
- „nuogas“ branduolys (*bare nuclei*);
- švelnus chromatinas (*bland chromatin*);
- normalus (nepakitęs) branduolėlis (*normal nucleoli*);
- mitozės (*mitoses*).

Sudaryti 9-mačiai ($n = 9$) vektoriai $X_1, X_2, \ldots, X_{683}$, atitinkantys 683 pacientus. Vektoriai $X_i = (x_{i1}, x_{i2}, \ldots, x_{i9})$, $i = 1, \ldots, 683$, yra priskiriami vienai iš dviejų žinomų klasių (nepiktybinis ar piktybinis navikas).

2. Daugiamačių duomenų vizualizavimo strategijos

Šiame skyriuje pateikiama daugiamačių duomenų vizualizavimo metodų apžvalga ir metodų taikymo pavyzdžiai. Nagrinėjamos kelios daugiamačių duomenų vizualizavimo strategijos (žr. 2.1 pav.):

- *tiesioginio vizualizavimo metodai*, kuriais kiekvienas daugiamačio objekto parametras yra pateikiamas tam tikra vizualia forma;
- *projekcijos,* dar vadinamieji *matmenų skaičiaus mažinimo* (*dimension reduction*) *metodai*, leidžiantys daugiamačius duomenų objektus atitinkančius vektorius pateikti mažesnio skaičiaus matmenų erdvėje; *dirbtiniai neuroniniai tinklai* (DNT, *artificial neural network*, ANN) taip pat gali būti naudojami daugiamačių duomenų projekcijoms rasti.

Daugiamačių duomenų *tiesioginio vizualizavimo metoduose* nėra apibrėžto formalaus matematinio vizualizavimo kokybės kriterijaus. Visi daugiamačius objektus apibūdinantys parametrai pateikiami žmogui priimtina vizualia forma. Šiuos metodus galima suklasifikuoti į geometrinius, simbolinius ir hierarchinio vizualizavimo.

1. Geometriniai metodai (*geometric techniques*):
   1.1. Taškinių grafikų (*scatter plots*).
   1.2. Taškinių grafikų matricos (*matrix of scatter plots*).
   1.3. Linijinių grafikų (*line graphs, multiline graphs*).
   1.4. Perstatymų matricos (*permutation matrix*).
   1.5. Apžiūros grafikų (*survey plots*).
   1.6. Andrews kreivių (*Andrews curves*).
   1.7. Lygiagrečiųjų koordinačių (*paralell coordinates*).
   1.8. Spindulinio vizualizavimo (*RadViz*), jo modifikacijų.
2. Simboliniai metodai (*iconographic display*):
   2.1. Černovo veidų (*Chernoff faces*).
   2.2. Žvaigždžių (*star glyphs*).
   2.3. Brūkšnelinės figūros (*stick figure*).
   2.4. Spalvotos piktogramos (*color icon*).
3. Hierarchinio vizualizavimo metodai (*hierarchical display*):
   3.1. Matmenų įterpimo (*dimensional stacking*).
   3.2. Grotelių (*trellis display*).
   3.3. Fraktalų (*fractal foam*).

3.4. Hierarchinių lygiagrečiųjų koordinačių (*hierarchical parallel coordinates*).

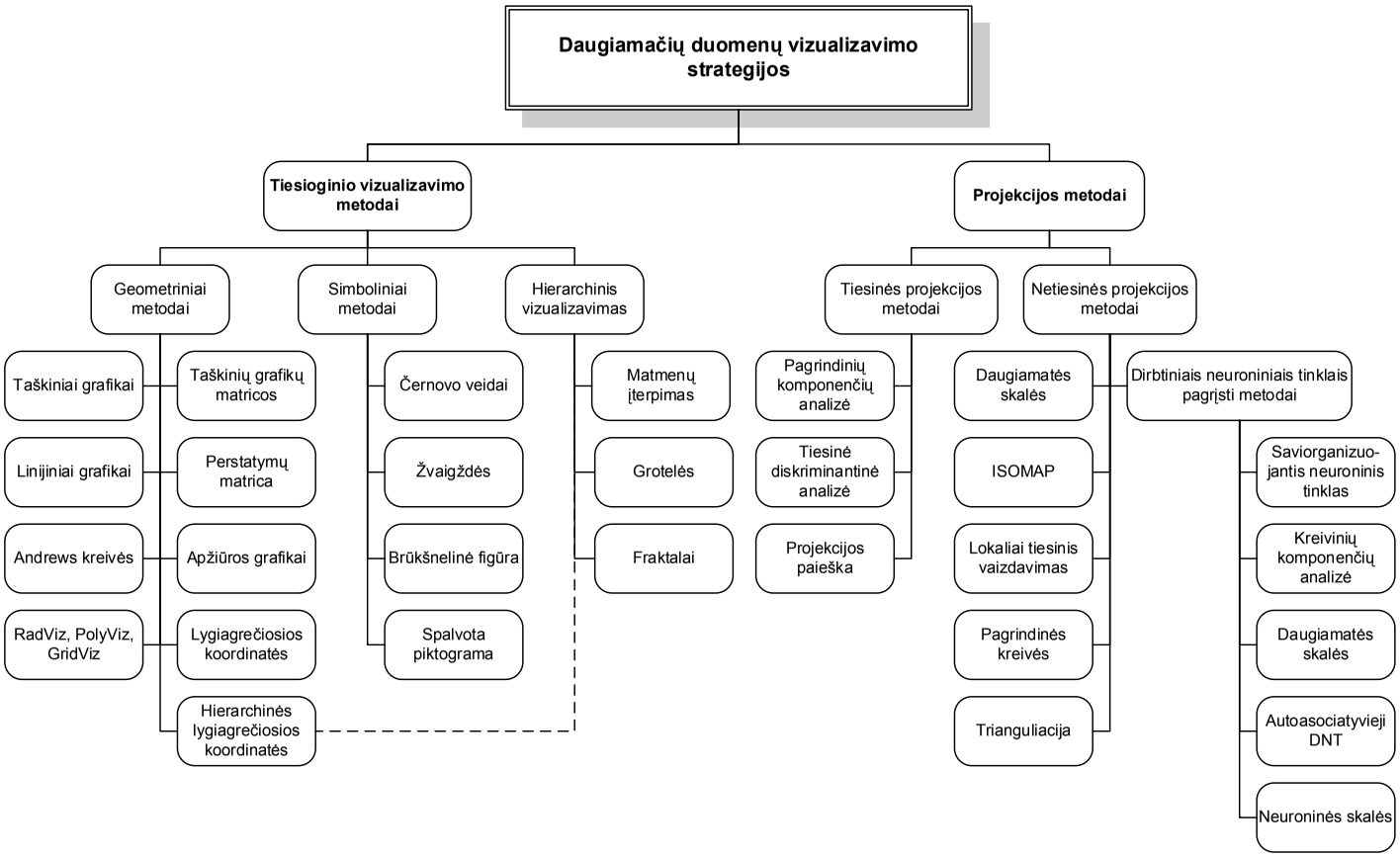
Metodai, leidžiantys pateikti daugiamačius duomenis mažesnio skaičiaus matmenų erdvėje, vadinami *projekcijos*, *arba matmenų mažinimo*, *metodais*. Kai pasirenkamas pakankamai mažas projekcinės (vaizdo) erdvės matmenų skaičius ($d = 2$ ar $d = 3$), šie metodai gali būti naudojami daugiamačiams duomenims vizualizuoti. Projekcijos metodai remiasi formaliais matematiniais kriterijais, pagal kuriuos minimizuojamas projekcijos iškraipymas. Dažniausiai išskiriami tiesinės ir netiesinės projekcijos metodai.

1. Tiesinės projekcijos metodai:
   1.1. Pagrindinių komponenčių analizės (PKA, *principal component analysis*, PCA).
   1.2. Tiesinės diskriminantinės analizės (TDA, *linear discriminant analysis*, LDA).
   1.3. Projekcijos paieškos (*projection pursuit*).
2. Netiesinės projekcijos metodai:
   2.1. Daugiamačių skalių (DS, *multidimensional scaling*, MDS).
   2.2. ISOMAP.
   2.3. Lokaliai tiesinio vaizdavimo (*locally linear embedding*, LLE).
   2.4. Pagrindinės kreivės (*principal curves*).
   2.5. Trianguliacijos (*triangulation*).

Dirbtiniai neuroniniai tinklai taip pat gali būti naudojami daugiamačiams duomenims vizualizuoti. Jie realizuoja įvairias netiesines projekcijas. Šiems metodams skirtas visas 4 skyrius. Analizuojami šie metodai:

1. Saviorganizuojantis neuroninis tinklas (*self organizing map*, SOM). Taip pat SOM tinklo ir Sammono algoritmo junginiai.
2. Kreivinių komponenčių analizė (KKA, *curvilinear component analysis*).
3. Daugiamatės skalės taikant dirbtinius neuroninius tinklus.
   3.1. Mokymo su mokytoju strategija.
   3.2. Mokymo be mokytojo strategija (SAMANN).
4. Autoasociatyvieji neuroniniai tinklai (*autoassociative neural network*).
5. Neuroninių skalių metodas (*neural scales*).

Pateikėme vieną iš galimų daugiamačių duomenų vizualizavimo strategijų klasifikacijų. Kitas vizualizavimo metodų apžvalgas bei klasifikacijas galima rasti šiuose darbuose: [58], [59], [73], [84], [85], [127], [132], [147], [150].



2.1 pav. Daugiamačių duomenų vizualizavimo strategijos

2.1. Tiesioginio vizualizavimo metodai

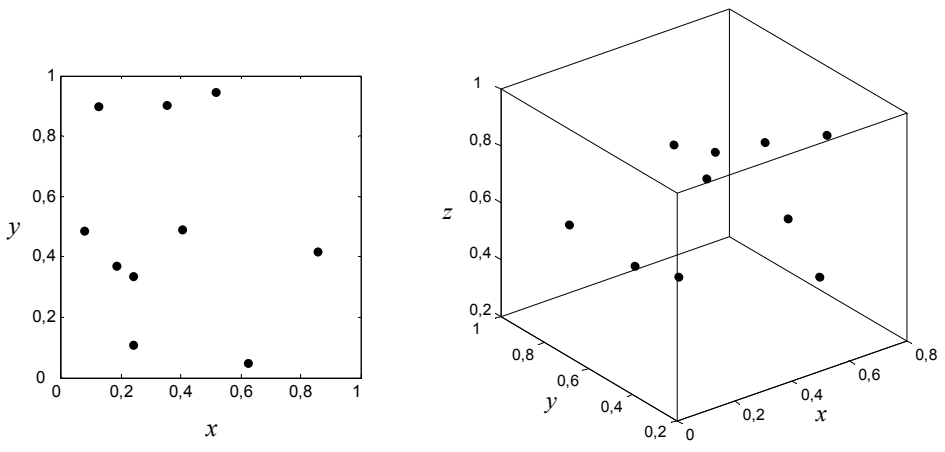
Nagrinėjami šiame skyriuje daugiamačių duomenų vizualizavimo metodai neturi apibrėžto formalaus matematinio vizualizavimo kokybės kriterijaus. Kiekvienas iš daugiamatį objektą X_i, $i \in \{1,\ldots,m\}$, apibūdinančių parametrų $x_1, x_2, \ldots, x_n$ pateikiamas žmogui priimtina vizualia forma.

2.1.1. Geometriniai metodai

Geometriniai vizualizavimo metodai yra tokie, kuriuose analizuojamus daugiamačius duomenis atitinkančių vektorių komponenčių reikšmės yra vaizduojamos naudojant pasirinktos geometrinės figūros ašis. Pagrindinė tokių metodų idėja – vizualizuoti duomenų transformacijas Dekarto arba kitoje koordinačių sistemoje [127].

Taškiniai grafikai

Taškiniai grafikai (*scatter plots*) yra vienas dažniausiai naudojamų duomenų pateikimo plokštumoje R^2 arba trimatėje erdvėje R^3 būdų. Taškai (vektoriai) atvaizduojami įprastu (x, y) arba (x, y, z) formatu [58], [59], [73]. Dažniausiai šiuo būdu atvaizduojami dvimačiai ($n = 2$) arba trimačiai ($n = 3$) taškai (žr. 2.2 pav.).

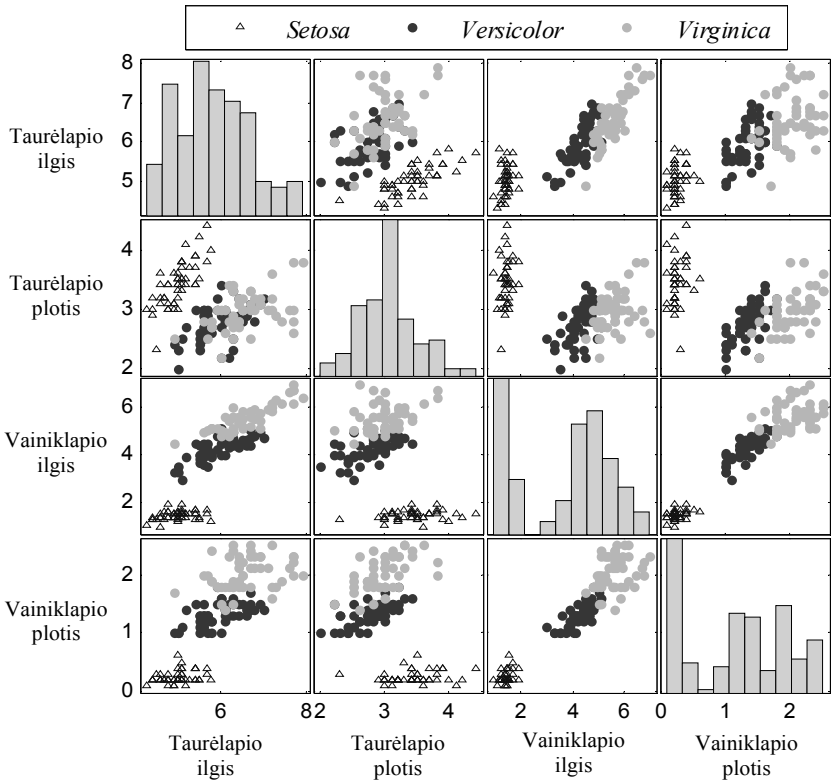


2.2 pav. Taškinių grafikų pavyzdžiai: a) dvimačiai taškai plokštumoje, b) trimačiai taškai erdvėje

Taškinių grafikų matrica

Naudojant taškinių grafikų matricą, taškinius grafikus galima taikyti daugiau nei trijų matmenų duomenims vizualizuoti. Tokioje matricoje atvaizduojamos visos galimos *n*-mačius duomenis apibūdinančių parametrų poros [85]. Tokių skirtingų porų yra $n(n-1)/2$. Matricos įstrižainėje gali būti pateikiama viena iš grafinių statistikinių parametro išraiškų, pavyzdžiui, histograma (žr. 2.3 pav.). Taškinių grafikų matricos yra naudingos ieškant koreliacijų tarp dviejų parametrų. Irisų duomenų taškinių grafikų matrica pavaizduota 2.3 paveiksle (duomenų aprašą žr. 1.4 skyrelyje). Iš paveikslo galima matyti, pagal kuriuos du parametrus irisų veislės labiau atsiskiria.

Taškiniai grafikai nebūtinai pateikiami stačiakampio formos – gali būti ir skritulio, šešiakampio ar kitokios figūros. Taip pat gali būti sukurtos įvairios matricų kombinacijos [73].



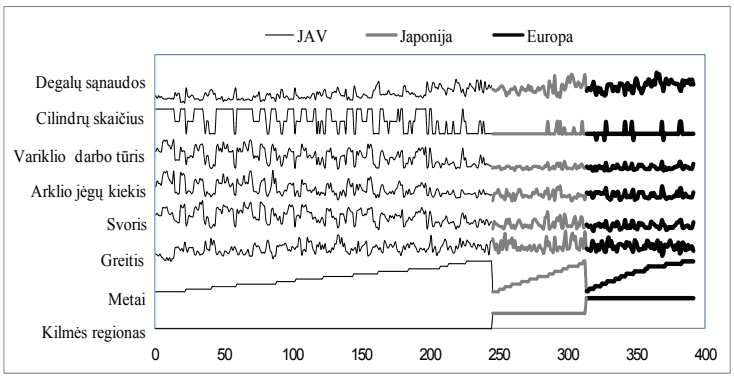
2.3 pav. Taškinių grafikų matrica (vizualizuoti irisų duomenys)

Linijiniai grafikai

Linijiniai grafikai (*line graphs*) yra naudojami vieno kintamojo funkcijoms vaizduoti. Jie dažniausiai taikomi dvimačiams duomenims vizualizuoti [57], [73]. Linijiniai grafikai yra gaunami sujungus linija taškus, kurių vietą nurodo koordinatės x ir y. Grafike nubraižytas tinklelis padeda nustatyti taškų koordinačių reikšmes.

Turint daugiamačius duomenis (kai $n > 2$), būtina naudoti daugialinijinius grafikus (*multiline graphs*). Daugialinijiniuose grafikuose reikia apibrėžti vieną kintamąjį (pvz., koordinatę x). Tai gali būti vektorių eilės numeris, t. y. skaičiai nuo 1 iki m, čia m – analizuojamų vektorių skaičius. Daugialinijiniame grafike viena linija atitinka vienos vektoriaus komponentės reikšmes.

Daugialinijinio grafiko pavyzdys pateiktas 2.4 paveiksle. Čia vizualizuoti automobilių duomenys (jų aprašą žr. 1.4 skyrelyje). Pirmiausia duomenys surūšiuoti pagal automobilių kilmės regioną (JAV, Europa, Japonija), kiekvienos kilmės regiono grupėje – pagal pagaminimo metus. Čia ašyje x atidėti analizuojamų objektų (automobilių) eilės numeriai duomenų sekoje. Analizuojamus duomenis apibūdinančių parametrų reikšmės gali būti iš skirtingų intervalų, todėl prieš vizualizuojant patartina duomenis sunormuoti (detaliau apie duomenų normavimą žr. 1.3 skyrelyje). Ašyje y pavaizduotos sunormuotos skirtingų parametrų reikšmės. Grafikų linijos storis priklauso nuo automobilio kilmės regiono. Iš paveikslo matyti automobilius apibūdinančių parametrų reikšmių kitimas priklausomai nuo pagaminimo metų ir kilmės. Pavyzdžiui, automobilių, pagamintų JAV, vidutinis cilindrų skaičius yra didesnis nei pagamintų Japonijoje ar Europoje.

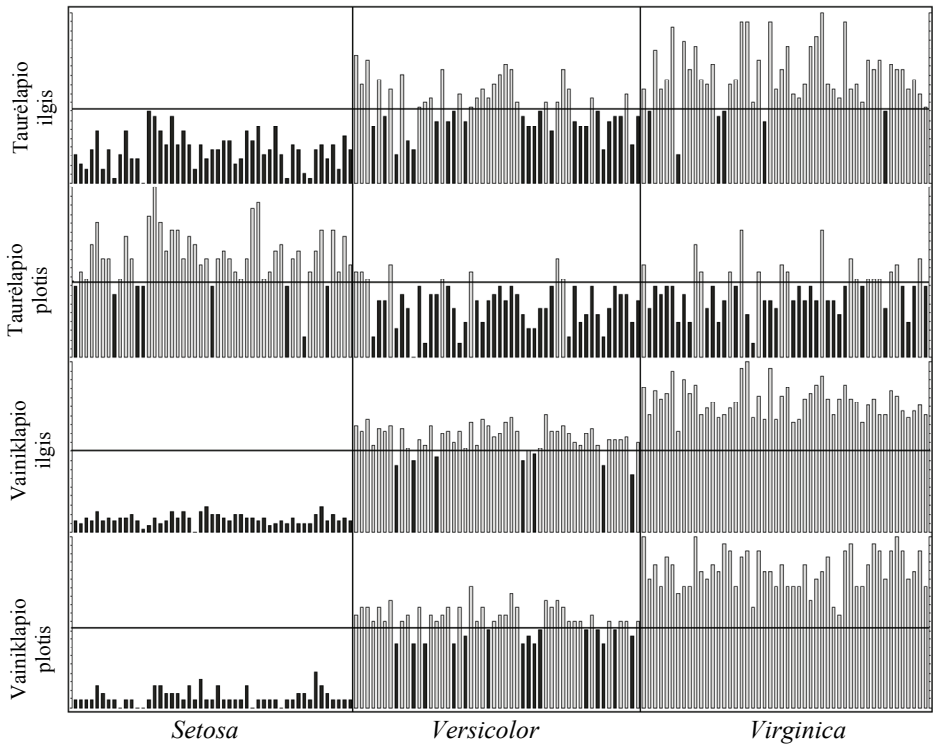


2.4 pav. Automobilių duomenų daugialinijinis grafikas

Perstatymų matrica

Džekas Bertinas 1960 metais pasiūlė perstatymų matricą (*permutation matrix*), kurios detalesnę analizę pateikė [12] darbe. Šis grafikas panašus į daugialinijinį grafiką. Čia stulpelių aukštis atitinka daugiamačius duomenis apibūdinančių parametrų reikšmes. Surūšiavus duomenis pagal vieną parametrą ar duomenų klases, galima stebėti, kaip keičiasi kitų parametrų išsidėstymas.

Perstatymų matricos grafikas, kuriame vizualizuoti irisų duomenys, pateiktas 2.5 paveiksle. Čia stulpelis, žymintis parametrą, kurio reikšmė yra didesnė už to parametro vidurkį, nuspalvinamas pilkai, o stulpelis, žymintis parametrą, kurio reikšmė yra mažesnė už vidurkį, nuspalvinamas juodai. Taip pat nubrėžiama linija, atitinkanti duomenų parametrų vidurkį (paveiksle juoda horizontali linija). Irisų veislės atskirtos vertikaliomis linijomis. Paveikslas gautas naudojantis *Visulab* sistema (http://www.inf.ethz.ch/personal/hinterbe/Visulab/).

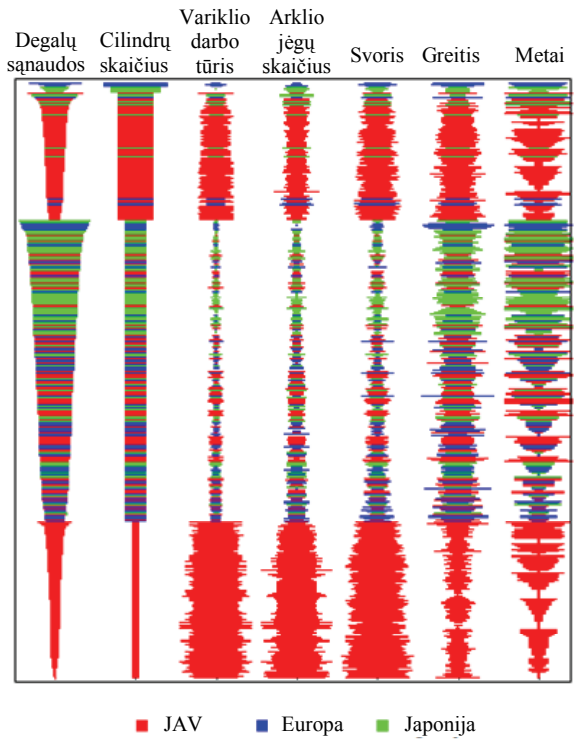


2.5 pav. Perstatymų matrica (vizualizuoti irisų duomenys)

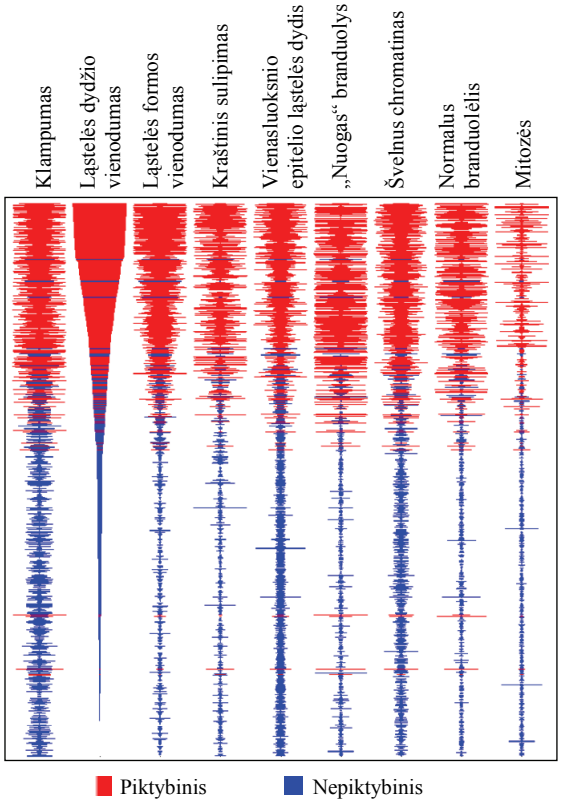
Apžiūros grafikai

Kaip ir perstatymų matricoje, apžiūros grafike (*survey plot*) kiekviena skaitinė parametro reikšmė vaizduojama linija, tiksliau sakant, juostele, kurios ilgis atitinka to parametro reikšmę. Perstatymų matrica yra pasukama 90° kampu, kiekviena parametrą atitinkanti juostelė perstumiama per pusę ilgio. Programoje *Inspect* [102] šis vizualizavimo metodas vadinamas apžiūros grafiku. Taip vizualiai pateikus *n*-mačius duomenis galima pamatyti koreliacijas tarp dviejų parametrų. Skirtingas klases nuspalvinus skirtingomis spalvomis, kartais galima pamatyti, kuris parametras yra reikšmingiausias klasifikuojant duomenis.

Apžiūros grafiku vizualizuoti automobilių duomenys pateikti 2.6 paveiksle, o krūties vėžio duomenys – 2.7 paveiksle. Grafikai gauti naudojantis *Orange* sistema (http://www.ailab.si/orange/).



2.6 pav. Apžiūros grafiku vizualizuoti automobilių duomenys, surūšiuoti pagal cilindrų skaičių ir degalų sąnaudas



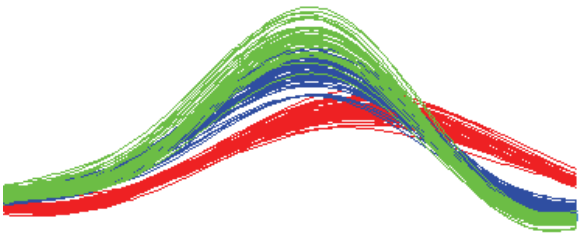
2.7 pav. Apžiūros grafiku vizualizuoti krūties vėžio duomenys, surūšiuoti pagal ląstelės dydžio vienodumo parametrą

Andrews kreivės

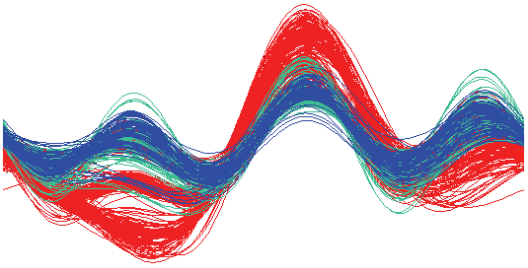
Vieną n-matį vektorių X_i, $i \in \{1,\ldots,m\}$, galima pavaizduoti *Andrews* kreive (sinusoidžių suma), kurios koeficientai yra analizuojamo vektoriaus X_i komponentės $x_{i1}, x_{i2}, \ldots, x_{in}$ [3].

$$f_i(t) = \frac{x_{i1}}{\sqrt{2}} + x_{i2}\sin(t) + x_{i3}\cos(t) + x_{i4}\sin(2t) + x_{i5}\cos(2t) + \cdots, \; -\pi < t < \pi .$$

Irisų ir automobilių duomenų Andrews kreives matome 2.8 paveiksle. Jos gautos naudojantis *Visulab* sistema (http://www.inf.ethz.ch/personal/hinterbe/Visulab/). Čia skirtingos irisų veislės (žr. 2.8a pav.) ir automobilių klasės pagal jų kilmės regioną (žr. 2.8b pav.) nuspalvintos skirtingomis spalvomis.



a)



b)

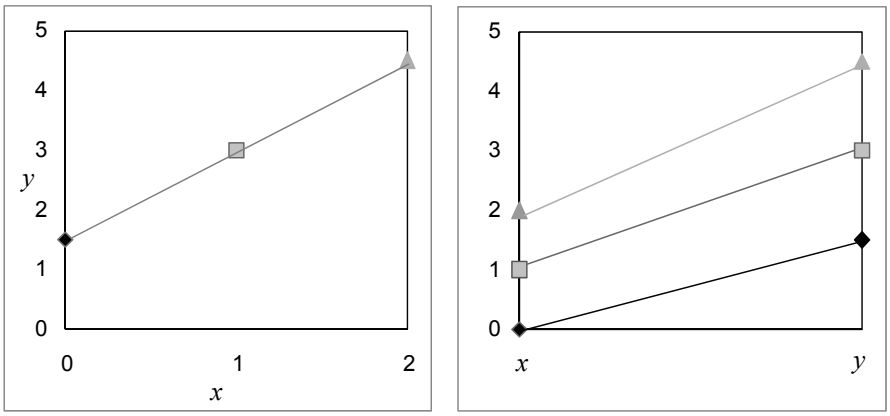
2.8 pav. Andrews kreivės: a) vizualizuoti irisų duomenys, b) vizualizuoti automobilių duomenys

Tos pačios veislės irisų kreivės yra viena šalia kitos, sudaro pluoštą. Vienos veislės kreivės atsiskiria nuo kitų dviejų. Automobilių duomenų kreivės yra labiau susimaišiusios ir jeigu klasės nebūtų išskirtos skirtingomis spalvomis, jas atskirti būtų neįmanoma (žr. 2.8b pav.). Vienas šio vizualizavimo metodo privalumų yra tai, kad metodą galima taikyti gana didelio skaičiaus matmenų duomenų analizei; trūkumas – vizualizavus didelės aibės duomenis, t. y. kai *m* gana didelis skaičius, juos suvokti ir interpretuoti yra gana sunku.

Lygiagrečiosios koordinatės

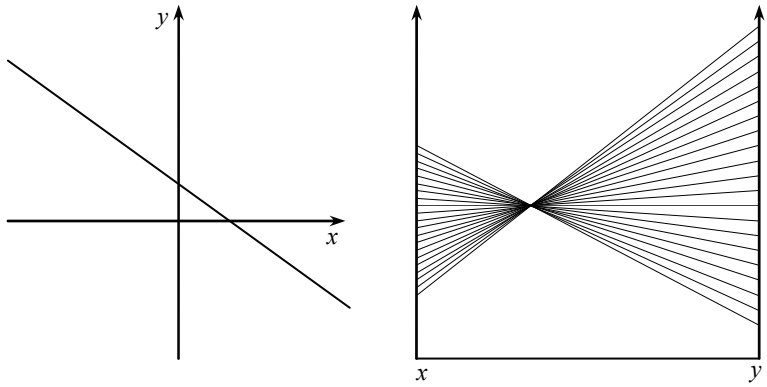
Lygiagrečiųjų koordinačių metodą (*parallel coordinates*), kaip naują būdą daugiamačiams duomenims vizualizuoti, pasiūlė A. Inselbergas 1981 metais [77]. Dekarto koordinačių sistemoje visos ašys yra statmenos viena kitai. Lygiagrečiųjų koordinačių sistemoje visos ašys yra lygiagrečios viena su kita ir išdėstytos vienodu atstumu [148] (žr. 2.9 pav.). Lygiagrečiosiose koordinatėse galima vaizduoti taškus, linijas, hiperplokštumas, kurių matmenų skaičius didesnis nei trys.

Taškus, esančius Dekarto koordinačių sistemos tiesėje $y = ax + b$ ($a \neq 1$), atidėjus lygiagrečiųjų koordinačių sistemoje, gaunamas pluoštas tiesių, susikertančių viename taške $\left(1/(1-a);\ b/(1-a)\right)$. Tai pavaizduota 2.10 paveiksle. Kai $a = 1$, nėra tinkamo būdo pavaizduoti tokią tiesę lygiagrečiųjų koordinačių sistemoje, nes tuo atveju gautume lygiagrečių tiesių aibę, kuri užpildytų visą koordinačių sistemą. Tradicinės Dekarto koordinačių sistemos dvimatis taškas lygiagrečiųjų koordinačių sistemoje virsta tiese, tiesė – tašku.



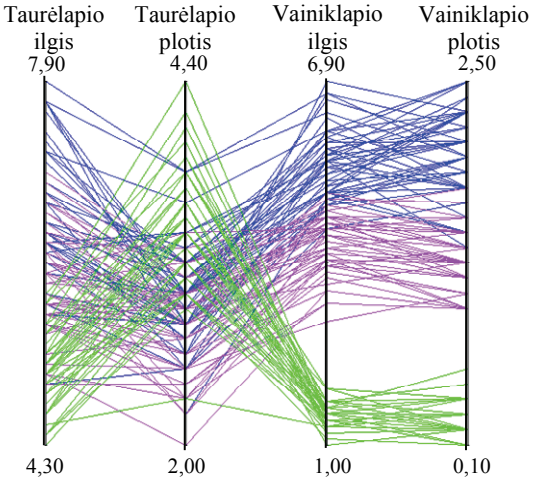
a) b)

2.9 pav. Trys dvimačiai taškai (0; 1,5), (1; 3), (2; 4,5), atidėti: a) Dekarto koordinačių sistemoje, b) lygiagrečiųjų koordinačių sistemoje



2.10 pav. Tiesės atvaizdavimas į tiesių pluoštą

Kaip atrodo irisų duomenys lygiagrečiosiose koordinatėse, parodyta 2.11 paveiksle. Skirtingos spalvos atitinka skirtingas irisų veisles. Kaip matome, geriausiai veislės atsiskiria pagal vainiklapio ilgį ir vainiklapio plotį. Pagal taurėlapio ilgį ir taurėlapio plotį veislių atskirti neįmanoma.



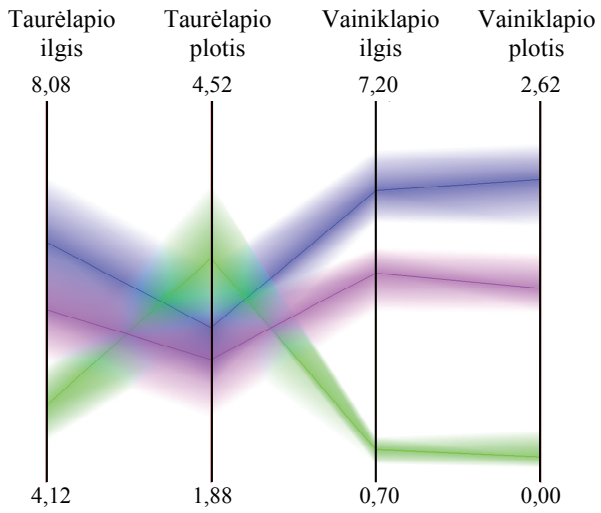
2.11 pav. Irisų duomenys, pavaizduoti lygiagrečiosiose koordinatėse

Lygiagrečiųjų koordinačių metodas gali būti taikomas gana didelio skaičiaus matmenų duomenims vizualizuoti. Tačiau tada tenka koordinates išdėstyti labai arti viena kitos. Taip sutankinus koordinates, sunku suvokti vizualizuotų duomenų struktūrą. Atvaizdavus didelę duomenų aibę, t. y. kai objektų skaičius *m* yra didelis, suvokti gautus rezultatus tampa taip pat labai sudėtinga, dažnai beveik neįmanoma. Lygiagrečiosios koordinatės, o ypač hierarchinės lygiagrečiosios koordinatės, gali būti naudojamos duomenims klasterizuoti.

Hierarchinės lygiagrečiosios koordinatės

Viena lygiagrečiųjų koordinačių metodo modifikacija yra hierarchinių lygiagrečiųjų koordinačių metodas, kurį pasiūlė Y. Fua, M. Wardas ir E. Rundensteineris [54]. Kai vizualizuojama didelė duomenų aibė hierarchinių lygiagrečiųjų koordinačių metodu, sumažėja persidengiančių linijų, gaunamų lygiagrečiųjų koordinačių metodu. Duomenys hierarchinėse lygiagrečiosiose koordinatėse pavaizduojami šiuo būdu:

- iš pradžių analizuojami duomenys yra suklasterizuojami į atskirus klasterius, naudojantis vienu iš klasterizavimo metodų [35], [66];
- duomenys pavaizduojami lygiagrečiosiose koordinatėse, išryškinami klasterių centrai; klasterių narių spalvos intensyvumas priklauso nuo to, kaip arti jie yra iki klasterio centro; skirtingi klasteriai vaizduojami skirtingomis spalvomis (žr. 2.12 pav.).

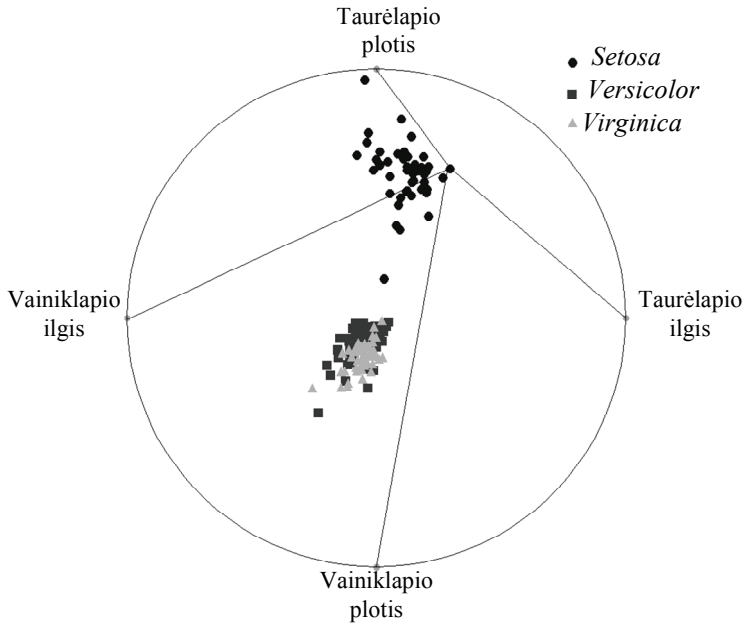


2.12 pav. Irisų duomenys, pavaizduoti hierarchinėse lygiagrečiosiose koordinatėse

Spindulinis vizualizavimas, jo modifikacijos

Spindulinio vizualizavimo metodas (*radial visualization*, *RadViz*) ir jo modifikacijos *PolyViz* ir *GridViz* sukurti Masačiusetso universiteto vizualizavimo institute (http://www.cs.uml.edu/~haim/ivpr.shtml). Iš skritulio centro nubrėžiama *n* spindulių (tiek, kiek yra parametrų, nusakančių analizuojamų objektų savybes). Kampai tarp spindulių turi būti vienodi, t. y. skritulio lankas padalijamas į *n* lygių dalių. Taškai, kuriuose spinduliai susikerta su skritulio lanku, vadinami inkarais (*dimensional anchors*), 2.13 paveiksle tai taurėlapio ilgis, taurėlapio plotis, vainiklapio ilgis, vainiklapio plotis [74]. Duomenims vizualizuoti taikoma spyruoklės veikimo paradigma. Vieni spyruoklių galai tarsi pritvirtinami prie kiekvieno inkaro, kiti galai – prie vizualizuojamo duomenų taško. Vieno taško spyruoklių sistema pavaizduota 2.13 paveiksle. Spyruoklių konstantos yra prilyginamos taško,

kuris yra vizualizuojamas, komponenčių reikšmėms. Prieš tai kiekvienos komponentės reikšmės turi būti sunormuotos intervale [0; 1] taip, kad mažiausia kiekvienos komponentės reikšmė būtų lygi nuliui, o didžiausia – vienetui (detaliau apie duomenų normavimą žr. 1.3 skyrelyje). Duomenų taškas atidedamas tame skritulio taške, kuriame visų spyruoklių jėgų suma lygi nuliui [15]. Paveikslas gautas naudojantis duomenų analizės sistema *Orange* (http://www.ailab.si/orange/).



2.13 pav. Spindulinio vizualizavimo metodu vizualizuoti irisų duomenys

Tegul turime daugiamačių duomenų rinkinį $X = \{X_1, X_2, \ldots, X_m\}$, čia $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, n – duomenis apibūdinančių parametrų skaičius. Norime juos vizualizuoti *Radviz* metodu, t. y. gauti dvimačius taškus $Y_1, Y_2, \ldots, Y_m$, čia $Y_i = (y_{i1}, y_{i2})$, $i = 1, \ldots, m$. Pažymėkime inkarus $S_1, S_2, \ldots S_n$, $S_j = (s_{j1}, s_{j2})$, $j = 1, \ldots, n$. Išsprendus lygtį

$$\sum_{j=1}^{n} (S_j - Y_i) x_{ij} = 0,$$

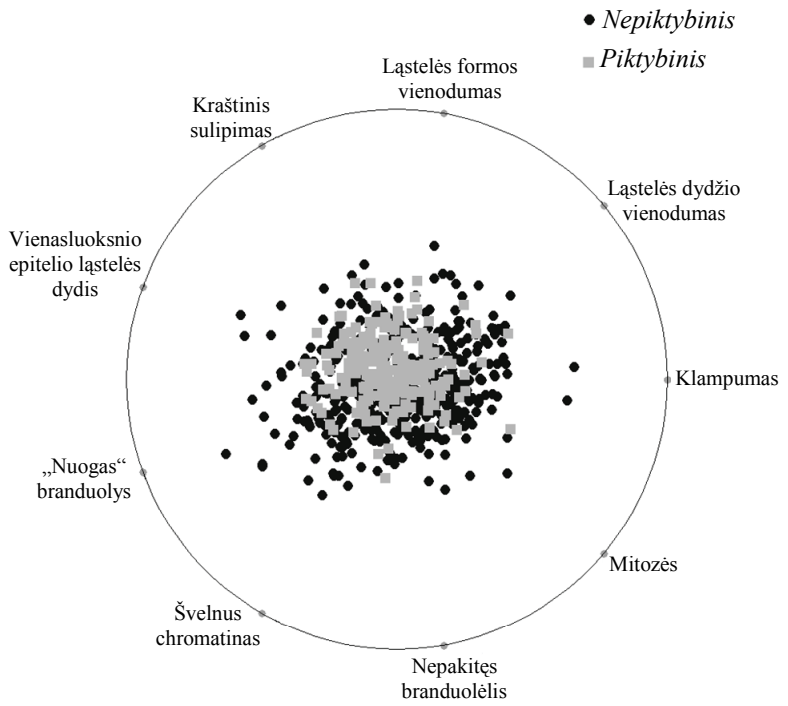
randamas taškas Y_i:

$$Y_i = \frac{\sum_{j=1}^{n} S_j x_{ij}}{\sum_{j=1}^{n} x_{ij}}.$$

RadViz metodo savybės:

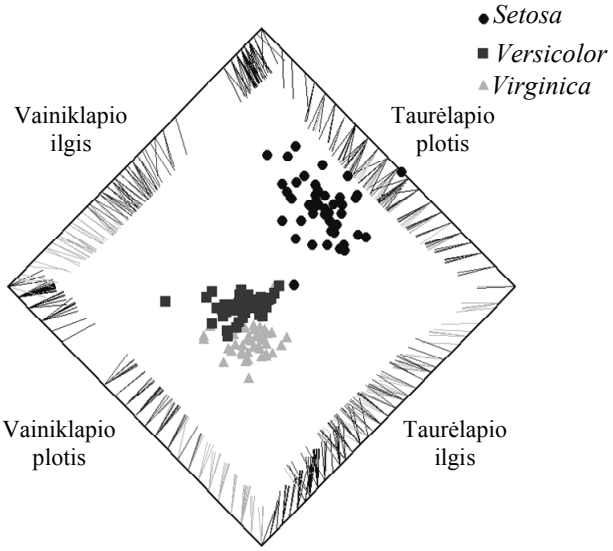
- daugiamačiai vektoriai, kurių visų komponenčių reikšmės tarpusavyje yra beveik lygios (po normavimo), pavaizduojami arti skritulio centro;
- daugiamačiai vektoriai, kurių komponentės su inkarais priešingose skritulio pusėse yra beveik lygios, pavaizduojami arti skritulio centro;
- daugiamačiai vektoriai, kurių vienos komponentės reikšmės yra daug didesnės už kitų, pavaizduojami arti tą komponentę atitinkančio inkaro;
- taškų vieta priklauso nuo inkarų išdėstymo ant skritulio lanko eiliškumo.

RadViz metodu vizualizuotus krūties vėžio duomenis matome 2.14 paveiksle. Dauguma piktybinio naviko atvejų koncentruojasi centre, tačiau vienareikšmiškai atskirti jų nuo nepiktybinio naviko atvejų beveik neįmanoma.



2.14 pav. Spindulinio vizualizavimo metodu vizualizuoti krūties vėžio duomenys

Yra sukurtos kelios *RadViz* metodo modifikacijos: *GridViz* (*grid visualization*) ir *PolyViz* (*polygon visualization*) [73]. *GridViz* atveju spyruoklių fiksuoti taškai (inkarai) atidedami ne ant skritulio lanko, kaip kad daroma *RadViz* metode, o stačiakampiame tinklelyje. Spyruoklių jėgos atlieka tą patį vaidmenį kaip *RadViz*: duomenų taškas atidedamas ten, kur visų spyruoklių jėgų suma lygi nuliui. *GridViz* atveju analizuojamus duomenis atitinkančių komponenčių pavadinimus sužymėti tinklelyje yra gana sunku, tačiau komponenčių skaičius gali būti didesnis nei taikant *RadViz* metodą. Naudojantis *RadViz* metodu tenka apsiriboti 50–100 komponenčių, o *GridViz* metodu galima lengvai pasinaudoti 50x50 (2500) tinkleliu. Vienas iš *RadViz* metodo trūkumų yra ir tai, kad gana skirtingi *n*-mačiai vektoriai (taškai) gali patekti į tą patį skritulio tašką. Jei komponentes atitinkantys inkarai būtų ne taškai, o atkarpos, taškų persidengimo problema sumažėtų. Tas atkarpas vadinsime inkarų atkarpomis. Tuo tikslu buvo sukurtas *PolyViz* (daugiakampio vizualizavimo) metodas (žr. 2.15 pav.). Brūkšnelis paveiksle prasideda nuo konkretaus parametro reikšmę atitinkančio taško ant inkaro atkarpos ir yra nukreiptas į *n*-matį tašką atitinkantį tašką plokštumoje. Kaip ir spindulinio vizualizavimo metodu, taško plokštumoje koordinatės randamos išsprendus tam tikrą lygtį [73]. Analizuojant didelio skaičiaus matmenų duomenis, daugiakampis tampa panašus į apskritimą, o *PolyViz* metodas – į *RadViz*.



2.15 pav. *PolyViz* metodu vizualizuoti irisų duomenys

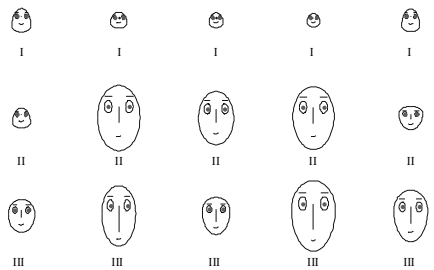
2.1.2. Simboliniai metodai

Daugiamačių duomenų vizualizavimo tikslas yra ne tik atvaizduoti duomenis dvimatėje ar trimatėje erdvėje, bet ir padėti lengviau juos suvokti. Būtent antrąjį tikslą galima bandyti pasiekti daugiamačius duomenis vizualizuojant simboliniais metodais (*iconographic display*). Šiais metodais kiekvienas analizuojamas objektas, kurį nusako *n*-matis vektorius, vaizduojamas pasirinktu simboliniu ženklu (*glyph*). Simbolio spalva, forma ar padėtis priklauso nuo analizuojamų vektorių komponenčių reikšmių.

Galima išskirti du simbolinių metodų tipus. Pirmojo tipo metodais analizuojami objektai iš tikrųjų vaizduojami tam tikru simboliu. Prie šio tipo priskiriami tokie metodai kaip Černovo veidai [20], žvaigždžių metodas [57], [82] ir kitų sudėtingesnių ženklų metodai [49], [92], [124], [149]. Antrojo tipo metodais simboliai suspaudžiami į tankų vaizdą ir gaunama tekstūra [73].

Černovo veidai

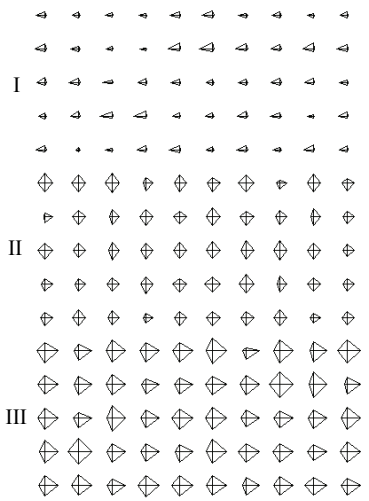
Tai daugiamačių duomenų vizualizavimo metodas, kurį sukūrė statistikas H. Černovas [20]. Vizualizuojamas objektas vaizduojamas stilizuotu veidu (žr. 2.16 pav.). Kiekvienas iš objektą nusakančių parametrų apibūdina tam tikros veido dalies dydį, padėtį ar formą, pavyzdžiui, vienas parametras – burnos plotį, kitas – akių formą ir pan. Pirmieji penki kiekvienos klasės irisai, vizualizuoti Černovo veidų metodu, pateikti 2.16 paveiksle. Čia taurėlapių ilgių reikšmės atitinka veido dydį, taurėlapių pločių – kaktos formą, vainiklapių ilgių – smakro formą, vainiklapių pločių – nosies ilgį. Kaip matome, veidai, atitinkantys I klasę (*Setosa*), labai skiriasi nuo veidų, atitinkančių II ir III klases (*Versicolor* ir *Virginica*). Veidai, atitinkantys II klasę (*Versicolor*), mažai skiriasi nuo veidų, atitinkančių III klasę (*Virginica*). Čia naudotasi *Matlab* sistema (http://www.mathworks.com).



2.16 pav. Černovo veidų metodu vizualizuoti irisų duomenų poaibis

Žvaigždės

Daugiamačiams duomenims vizualizuoti gali būti naudojamas žvaigždžių metodas (*star glyphs*) [57], [82]. Kiekvienas vizualizuojamas objektas vaizduojamas stilizuota žvaigžde. Iš vieno taško nubraižoma tiek spindulių, kiek yra objektą apibūdinančių parametrų. Kampai tarp spindulių turi būti vienodi. Spindulio ilgis priklauso nuo jį atitinkančio parametro reikšmės. Išoriniai spindulių galai yra sujungiami linijomis. Žvaigždžių metodu vizualizuotus irisų duomenis matome 2.17 paveiksle. Čia I klasės irisus (*Setosa*) atitinkančios žvaigždės yra mažesnės už kitų dviejų (II ir III) klasių. Didžiausios žvaigždės atitinka III klasę (*Virginica*). Naudotasi *Matlab* sistema (http://www.mathworks.com). Žvaigždžių, kaip ir Černovo veidų, metodo problema – kaip prasmingai išdėstyti žvaigždes ar veidus vienus šalia kitų.



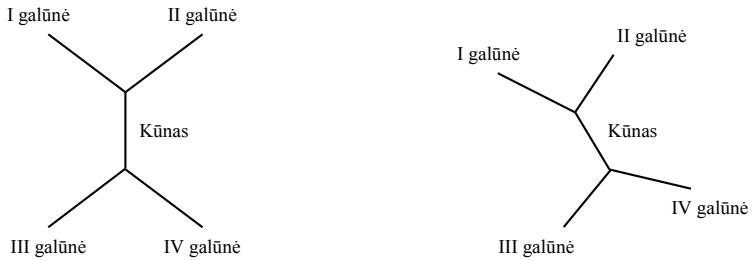
2.17 pav. Žvaigždžių metodu vizualizuoti irisų duomenys

Brūkšnelinė figūra

Dar vienas simbolinis yra brūkšnelinės figūros metodas (*stick figure icon*), pasiūlytas [120] darbe. Brūkšnelinė figūra susideda iš penkių sujungtų segmentų, vadinamų galūnėmis (*limbs*). Vienas segmentas vadinamas kūnu (*body*). Jis tarnauja kaip atrama įvairioms geometrinėms transformacijoms, kuriomis figūra gali pasikeisti priklausomai nuo vizualizuojamą objektą apibūdinančių parametrų reikšmių. Kiekviena galūnė turi kelias charakteristikas, kuriomis gali būti išreiškiami analizuojamus objektus

apibūdinantys parametrai. Tai galūnės ilgis, jos posūkio kampas, spalva ir kt. Pagal būdą, kuriuo galūnės prijungiamos prie kūno arba viena prie kitos, yra išskiriamos kelios brūkšnelinių figūrų šeimos [59], [80].

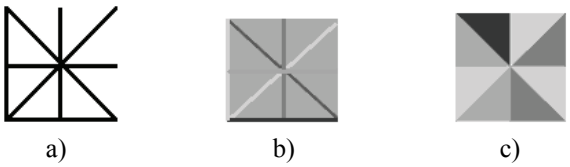
Bazinės konfigūracijos figūra, kai jokio vizualizuojamo objekto dar nėra, pateikta 2.18a paveiksle. Figūra, kurios galūnių parametrais pavaizduotas vienas vizualizuojamas objektas, matoma 2.18b paveiksle. Šioje figūroje visos keturios galūnės yra tiesiogiai sujungtos su kūnu. Brūkšnelinės figūros metodu vizualizuojant duomenų aibę, sudarytą iš didelio skaičiaus objektų, figūros suglaudžiamos vieną šalia kitos ir gaunama tekstūra [73].



2.18 pav. Brūkšnelinė figūra: a) bazinė figūra, b) atvaizduotas vienas objektas

Spalvota piktograma

Piktogramos spalva, dydis, plotas ir kitos savybės gali būti naudojamos vizualizuojant daugiamačius duomenis [101]. Daugiamatį objektą spalvota piktograma (*color icon*) galima pavaizduoti dviem būdais. Pirmas būdas: braižomos linijos, kurių kiekviena atitinka vieną iš objektą apibūdinančių parametrų (žr. 2.19a pav.); kiekvienai linijai spalvos intensyvumas nustatomas pagal tai, kokia yra parametro, kurį atitinka ši linija, reikšmė; viso ženklo fonas yra vienodas (žr. 2.19b pav.). Antras būdas: piktogramos atskirų dalių spalvos nustatomos pagal parametrų reikšmes (žr. 2.19c pav.).



2.19 pav. Spalvota piktograma, atvaizduojanti 6-matį objektą

2.1.3. Hierarchinio vizualizavimo metodai

Taikant hierarchinio vizualizavimo metodus (*hierarchical display*), vieni daugiamačius objektus apibūdinantys parametrai bandomi „įsprausti“ į kitus parametrus.

Matmenų įterpimas

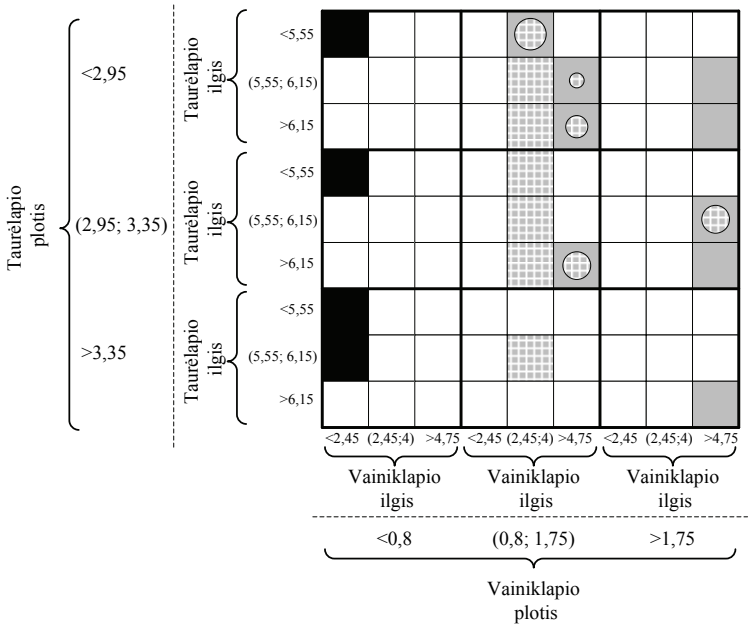
Šio metodo pirmtakas yra loginės diagramos [114]. Jose buvo pavaizduotos tik loginių duomenų reikšmės (0; 1). Vėliau M. Wardas [145], [146] praplėtė šį metodą. Matmenų įterpimo metodas (*dimensional stacking*) realizuotas *Xmdv* sistemoje (http://davis.wpi.edu/~xmdv/).

Matmenų įterpimo metodo schema:

- vizualizuojamus objektus apibūdinančių parametrų reikšmių kitimo intervalai suskaidomi į pointervalius; rekomenduojama, kad tokių pointervalių būtų ne daugiau kaip 5;
- dviejų pasirinktų parametrų, vadinamųjų išoriniais, pora atvaizduojama tinkleliu, kurio eilučių ir stulpelių skaičius priklauso nuo pointervalių skaičiaus; faktiškai tų dviejų parametrų reikšmių kitimo sritis padalijama į stačiakampius, kurie ir yra to tinklelio langeliai;
- vaizduojant kitus du parametrus, vadinamuosius vidiniais, kiekviename išorinių parametrų suformuoto tinklelio langelyje kuriamas naujas tinklelis, atvaizduojantis vidinių parametrų porą, t. y. vidinius parametrus vaizduojantys tinkleliai yra „įspraudžiami“ į kiekvieną išorinių parametrų reikšmių kitimo sritį padengiančio tinkelio langelį;
- toks tinklelių langelių padengimo smulkesniais tinkleliais procesas tęsiamas tol, kol „įspraudžiami“ visi parametrai;
- tinklelio langeliai, kurie nebepadengti smulkesniais tinkleliais, nuspalvinami, jeigu yra vizualizuojamų objektų, kuriuos apibūdinančių parametrų reikšmės patenka į šių langelių nusakomas reikšmes; jei yra žinomos vizualizuojamų objektų klasės, tinklelio langelio spalva parenkama priklausomai nuo į jį patekusio objekto klasės.

Matmenų įterpimo metodu vizualizuotus irisų duomenis matome 2.20 paveiksle. Iš paveikslo matome, kad vienos veislės irisai (juoda spalva) yra atskiriami nuo kitų dviejų. Kitos dvi veislės truputį susimaišo.

Matmenų įterpimo metodas gali būti naudojamas ieškant klasterių, taškų atsiskyrėlių. Tačiau daugiau nei 8 matmenų duomenis atvaizduoti gana sudėtinga ir rezultatai sunkiai suvokiami.

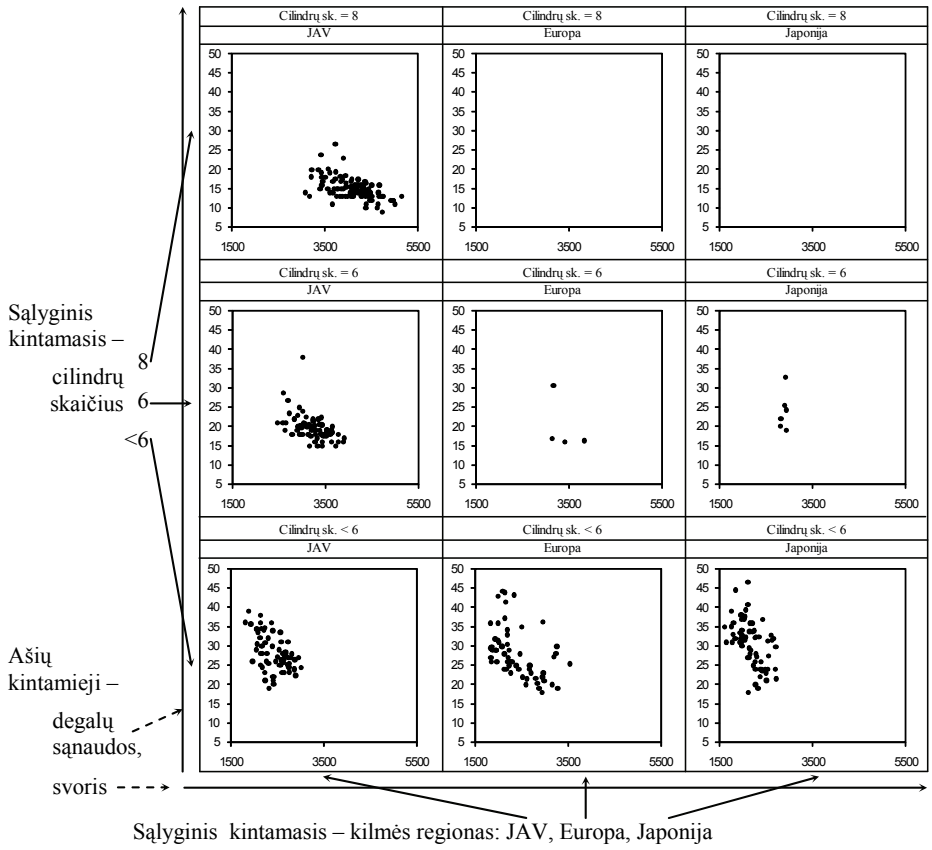


2.20 pav. Matmenų įterpimo metodu vizualizuoti irisų duomenys

Grotelės

Į matmenų įterpimo metodą labai panašus kitas, vadinamasis grotelių metodas (*trellis display*). Naudojantis šiuo metodu, taip pat yra galimybė atvaizduoti iki 8 matmenų duomenis [6]. Šio metodo angliškas pavadinimas *trellis display* kilo iš lotyniško žodžio *tre-liceum*, reiškiančio rėmelius, naudojamus vijokliniams lipantiems augalams. Grotelių metodu vizualizuotus automobilių duomenis matome 2.21 paveiksle.

Iš pradžių pasirenkami du vizualizuojamus objektus apibūdinantys parametrai. Jie vadinami ašių kintamaisiais (*axis variables*). Paveiksle tai – svoris ir degalų sąnaudos. Kiti parametrai vadinami sąlyginiais kintamaisiais (*conditioning variables*). Šių parametrų reikšmių kitimo intervalai yra padalijami į nesikertančius pointervalius. Sąlyginiais kintamaisiais taip pat gali būti vizualizuojamų objektų klasės, pavyzdžiui, kilmės regionas. Paveiksle sąlyginiai kintamieji yra kilmės regionas ir cilindrų skaičius, duomenys suskirstyti į tris intervalus: kai cilindrų skaičius mažiau už 6, lygus 6 arba 8. Kiekvienam pointervaliui ir (arba) klasei yra braižomi grafikai (*panel plot*), kurie gali būti taškiniai, stulpelių, paviršių ir kt. (paveiksle – taškiniai grafikai).



2.21 pav. Grotelių metodu vizualizuoti automobilių duomenys

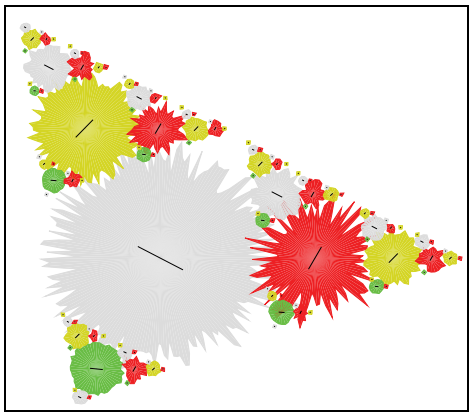
Fraktalai

Fraktalų metodas (*fractal foam*) yra naudojamas ne patiems daugiamačiams objektams, bet koreliacijoms tarp juos apibūdinančių parametrų ar kitoms statistikinėms savybėms vizualizuoti [122].

Fraktalų metodo schema:

- Pirmiausia vienas iš objektus apibūdinančių parametrų pasirenkamas kaip pradinis. Jis pavaizduojamas didžiausiu skrituliu paveikslo centre (žr. 2.22 pav.) ir vadinamas centriniu skrituliu (*focus bubble*).
- Kiti parametrai pavaizduojami mažesniais, vadinamaisiais aplinkiniais, skrituliais (*surrounding bubble*), „bėgančiais“ prieš laikrodžio rodyklę apie centrinį skritulį.

- Aplinkinių skritulių dydžiai priklauso nuo koreliacijų tarp pradinio parametro ir aplinkinius skritulius atitinkančių parametrų dydžio. Skirtingus parametrus atitinkantys skrituliai nuspalvinami skirtingomis spalvomis.
- Procesas kartojamas: vienoje iteracijoje kiekvienas prieš tai buvusios iteracijos metu pavaizduotas skritulys tampa centriniu skrituliu, apie jį braižomi aplinkiniai skrituliai, kurių dydžiai yra proporcingi koreliacijoms tarp centrinį ir aplinkinius skritulius atitinkančių parametrų. Skritulių braižymo procedūra kartojama tol, kol jie tampa labai maži.



2.22 pav. Fraktalų metodo taikymo pavyzdys irisų parametrams: taurėlapių ilgis (pilka), vainiklapių ilgis (raudona), vainiklapių plotis (geltona), taurėlapių plotis (žalia)

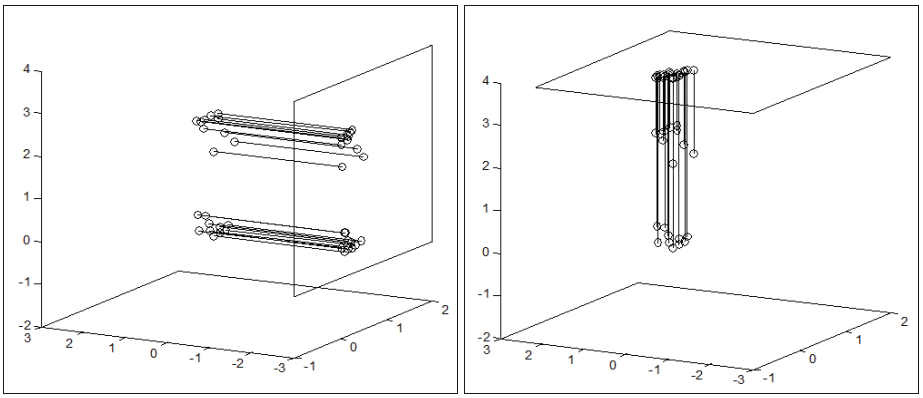
Kaip matome iš fraktalų metodo taikymo pavyzdžio skritulių forma nėra reguliari. Čia jie labiau primena krumpliaračius. Krumpliuotumas bendru atveju gali nurodyti statistikines duomenų savybes. Vietoje skritulių gali būti naudojamos ir sritys, apribotos elipsėmis.

Vizualizuotos koreliacijos tarp irisų duomenis apibūdinančių parametrų pateiktos 2.22 paveiksle. Čia pasinaudota *SPSS Diamond* sistema. Pilkos spalvos skrituliai žymi irisų taurėlapių ilgį, raudonos – vainiklapių ilgį, geltonos – vainiklapių plotį, žalios – taurėlapių plotį. Matome, kad koreliacija tarp taurėlapių ilgio ir vainiklapių ilgio, bei taurėlapių ilgio ir vainiklapių pločio yra didesnė nei tarp taurėlapių ilgio ir taurėlapių pločio. Taurėlapio plotis yra mažiausiai koreliuotas su kitais parametrais.

2.2. Projekcijos metodai

Analizuojant daugiamačius objektus (toliau juos dažnai vadinsime daugiamačiais duomenimis ar daugiamačiais taškais), kuriuos apibūdina n parametrų, norima vizualiai įvertinti jų išsidėstymą n-matėje erdvėje. Kai $n > 3$, tiesiogiai pamatyti šių taškų neįmanoma. Tačiau yra galimybė rasti jų projekciją į dvimatę ar trimatę erdvę, t. y. sumažinti matmenų skaičių iki mums įprasto.

Kaip trimačiai taškai projektuojami į plokštumą, parodyta 2.23 paveiksle. Dešinėje ir viršuje padėti ekranai. Įsivaizduokime, kad kitoje pusėje šviečia ryški šviesa. Ant ekranų matomi duomenų taškų šešėliai. Linijos sujungia taškus su jų projekcijomis. Iš 2.23a paveikslo matyti, kad dvimatėje projekcijoje išlieka du klasteriai, tačiau 2.23b paveiksle jokių klasterių negalima įžvelgti.



a) b)

2.23 pav. Trimačių taškų projekcijų pavyzdžiai

Iš šio pavyzdžio akivaizdu, kad skirtingomis projekcijomis gaunami skirtingi rezultatai. Be to, ne visada projekcijoje matome tikrąją duomenų struktūrą – klasterius, taškus atsiskyrėlius, taškų išsidėstymą. Todėl aukščiau minėto ekrano padėties pasirinkimo uždavinys nėra trivialus.

Vizualizuojant daugiamačius duomenis susiduriama su dviem dažnai vienas kitam prieštaraujančiais tikslais: viena vertus, norima supaprastinti uždavinį mažinant duomenų matmenų skaičių, kita vertus, norima išlaikyti kiek galima daugiau originalios informacijos turinio, t. y. kuo mažiau iškraipyti analizuojamus duomenis.

Daugiamačiams duomenims transformuoti į mažesnio skaičiaus matmenų erdvę taikomi *projekcijos metodai*. Jie dar vadinami *matmenų skaičiaus mažinimo metodais* (*dimensional reduction techniques*). Jų tikslas – pateikti daugiamačius duomenis mažesnio skaičiaus matmenų erdvėje taip, kad būtų kiek galima tiksliau išlaikyta duomenų struktūra.

Projekcijos metodai gali būti naudojami ir daugiamačiams duomenims vizualizuoti, kai pasirinktas pakankamai mažas projekcinės erdvės R^d matmenų skaičius ($d = 2$ arba $d = 3$). Erdvė R^d dar vadinama *vaizdo erdve*, nes dažnai jos elementus galima stebėti vizualiai.

Tarkime, kad analizuojamų duomenų aibė yra matrica

$$X = \{X_1, X_2, ..., X_m\} = \{x_{ij}, i = 1, \ldots, m, j = 1, \ldots, n\},$$

kurios *i*-oji eilutė yra vektorius $X_i \in R^n$, čia $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $i \in \{1, \ldots, m\}$, x_{ij} yra duomenų *i*-ojo vektoriaus X_i *j*-oji komponentė, atitinkanti *j*-ąjį parametrą, *n* – vektoriaus X_i komponenčių skaičius, t. y. erdvės, kuriai priklauso vektorius X_i, matmenų skaičius, *m* – analizuojamų objektų (vektorių) skaičius.

Reikia rasti vektoriaus $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$ transformaciją $Y_i = (y_{i1}, y_{i2}, \ldots, y_{id})$ mažesnio skaičiaus matmenų *projekcinėje*, arba *vaizdo*, erdvėje R^d, $d < n$. Jeigu projekcijos metodo tikslas – vizualizuoti duomenis, tai $d = 2$ arba $d = 3$. Galima nagrinėti ir atvejį, kai $d = 1$, tačiau tada vizualizuojant prarandama daugiau informacijos ir vaizdumo, turimo stebint taškus plokštumoje $(d = 2)$ ar erdvėje $(d = 3)$ lyginant su tų pačių taškų išsidėstymu tiesėje $(d = 1)$.

Projekcijos metoduose yra formalūs matematiniai projekcijos kokybės kriterijai, kurie optimizuojami siekiant gauti kiek galima tikslesnę daugiamačių duomenų projekciją mažesnio skaičiaus matmenų erdvėje, t. y. kuo mažiau iškraipyti duomenis pasirinkto kriterijaus atžvilgiu, išsaugoti vizualizuojamų daugiamačių vektorių tarpusavio atstumų ar kitų artimumo įverčių proporcijas, taip pat išsaugoti ar net išryškinti kitas sprendimams priimti svarbias analizuojamos duomenų aibės *X* charakteristikas, pavyzdžiui, klasterius.

Yra išskiriami *tiesinės* ir *netiesinės projekcijos* metodai. Tiesinės projekcijos metodais ieškoma tiesinės analizuojamų duomenų transformacijos, o netiesinės projekcijos – netiesinės transformacijos. Bendru atveju yra įvairių tiesinės transformacijos galimybių: pasukimas, postūmis, atspindys, suspaudimas ir kt.

Pasukimo tipo tiesinė transformacija yra aprašoma tiesinių lygčių sistema

$$Y_i = X_i A.$$

Bendru atveju $Y_i = (y_{i1}, y_{i2}, \ldots, y_{in})$, $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, A – kvadratinė matrica, sudaryta iš n eilučių ir n stulpelių. Jei tiesinė transformacija naudojama duomenų matmenų skaičiui mažinti, tada $Y_i = (y_{i1}, y_{i2}, \ldots, y_{id})$, $d < n$, A – matrica, sudaryta iš n eilučių ir d stulpelių.

Netiesinė projekcija yra aprašoma šia forma:

$$Y = f(X),$$

čia f yra netiesinė transformacija. Netiesinė projekcija yra sudėtingesnė, reikalauja kur kas didesnių skaičiavimų – čia formuluojami ir sprendžiami daugelio kintamųjų optimizavimo uždaviniai, kurie dažniausiai būna daugiaekstremiai.

Panagrinėkime elementarų tiesinės projekcijos atvejį, kai $n = 2$. Tegul turime tašką $X_i = (x_{i1}, x_{i2})$. Transformuokime jį į tašką $Y_i = (y_{i1}, y_{i2})$ tiesine transformacija

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}.$$

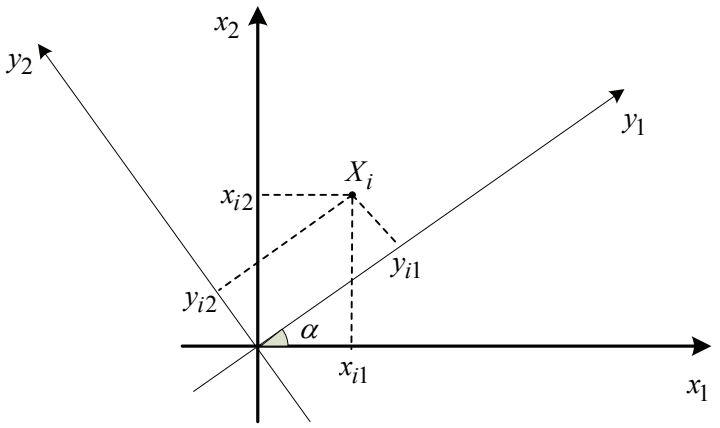
Matricos A elementai šiuo atveju priklauso tik nuo pasukimo kampo ir gali būti išreikšti per trigonometrines funkcijas:

$$A = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix},$$

čia α yra kampas tarp ašių x_1 ir y_1, tuo pačiu ir tarp ašių x_2 ir y_2, t. y. koordinačių sistemą (x_1, x_2) kampu α pasukame prieš laikrodžio rodyklę kad gautume koordinačių sistemą (y_1, y_2) (žr. 2.24 pav.). Koordinačių pradžios taškas nepasikeičia. Taško X_i koordinatės sistemoje (x_1, x_2) yra (x_{i1}, x_{i2}), o naujoje sistemoje yra (y_{i1}, y_{i2}):

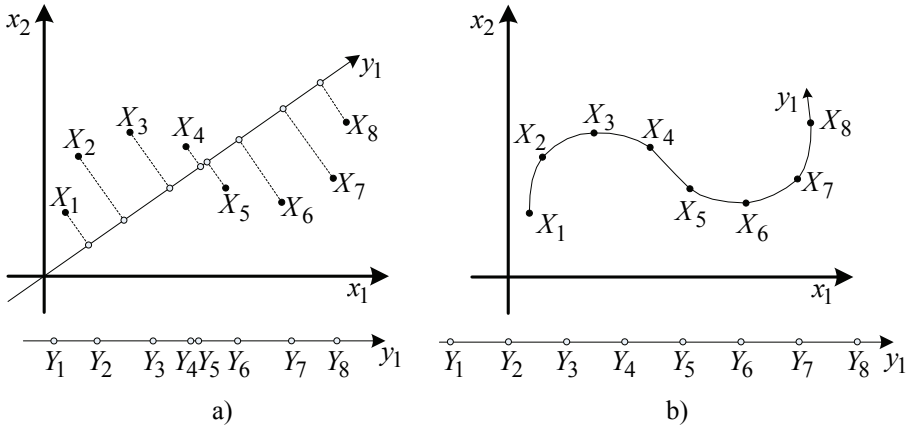
$$y_{i1} = x_{i1}\cos\alpha + x_{i2}\sin\alpha, \qquad y_{i2} = x_{i2}\cos\alpha - x_{i1}\sin\alpha.$$

Faktiškai $Y_i = (y_{i1}, y_{i2})$ ir yra taško X_i tiesinė transformacija į koordinačių sistemą (y_1, y_2). Jei mūsų tikslas yra sumažinti duomenų matmenų skaičių, tai vektorių Y_i galime sudaryti tik iš vienos koordinatės, pavyzdžiui, $Y_i = (y_{i1})$. Tada gausime taško X_i transformaciją į vienmatę erdvę.



2.24 pav. Tiesinės transformacijos (pasukimo) pavyzdys

Tiesinė ir netiesinė projekcijos pavaizduotos 2.25 paveiksle. Dvimačiai taškai $X_1, X_2, \ldots, X_8$ išdėstyti taip, kad tarp artimiausių taškų yra vienodi atstumai, t. y. $d(X_i, X_{i+1}) = d(X_{i+1}, X_{i+2})$, $i = 1, \ldots, 6$. Jei juos atvaizduosime į vienmatę erdvę (į tiesę y_1) naudodamiesi tiesine projekcija, atstumai tarp taškų nebus išlaikyti (žr. 2.25a pav.). Tačiau netiesinės projekcijos atveju, radus tinkamą transformaciją, atstumai tarp artimiausių taškų išliks vienodi (žr. 2.25b pav.).



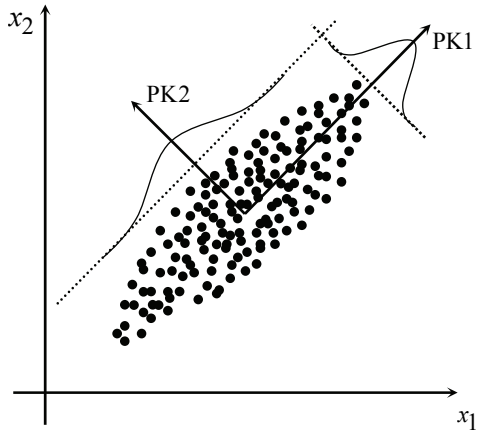
2.25 pav. Projekcijos: a) tiesinė ir b) netiesinė

2.2.1. Tiesinės projekcijos metodai

Pagrindinių komponenčių analizė

Pagrindinių komponenčių analizė (PKA, *principal component analysis*, PCA) yra klasikinis statistikos metodas. Tai tiesinė duomenų transformacija, plačiai naudojama duomenų analizei kaip daugiamačių duomenų matmenų skaičiaus mažinimo metodas.

Esminė PKA idėja yra sumažinti duomenų matmenų skaičių atliekant tiesinę transformaciją ir atsisakant dalies po transformacijos gautų naujų komponenčių, kurių dispersijos yra mažiausios [119] [135], [152]. Iš pradžių ieškoma krypties, kuria dispersija yra didžiausia. Didžiausią dispersiją turinti kryptis vadinama pirmąja pagrindine komponente (PK1, žr. 2.26 pav.). Ji eina per duomenų centrinį tašką. Tai taškas, kurio komponentės yra analizuojamą duomenų aibę sudarančių taškų atskirų komponenčių vidurkiai. Visų taškų vidutinis atstumas iki šios tiesės yra minimalus, t. y. ši tiesė yra kiek galima arčiau visų duomenų taškų. Antrosios pagrindinės komponentės (PK2) ašis taip pat turi eiti per duomenų centrinį tašką ir ji turi būti statmena pirmosios pagrindinės komponentės ašiai.



2.26 pav. Pirmoji (PK1) ir antroji (PK2) pagrindinės komponentės

Apibrėžkime sąvokas, kurios yra vartojamos pagrindinių komponenčių analizėje. Tegul turime duomenų matricą

$$X = \{X_1, X_2, \ldots, X_m\} = \{x_{ij},\ i = 1, \ldots, m,\ \ j = 1, \ldots, n\},$$

kurios *i*-oji eilutė yra vektorius $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$. Faktiškai šios matricos eilutės žymi objektus, o stulpeliai – tuos objektus apibūdinančius parametrus $x_1, x_2, \ldots, x_n$.

Apskaičiuokime *koreliacijos koeficientą*. *Koreliacija* – tai dviejų parametrų tarpusavio sąryšis. Parametrų x_k ir x_l koreliacijos koeficientas r_{kl} tarp yra skaičiuojamas pagal šią formulę:

$$r_{kl} = \frac{\sum_{i=1}^{m}(x_{ik} - \overline{x}_k)(x_{il} - \overline{x}_l)}{\sqrt{\sum_{i=1}^{m}(x_{ik} - \overline{x}_k)^2 \sum_{i=1}^{m}(x_{il} - \overline{x}_l)^2}}, \tag{2.1}$$

čia

$$\overline{x}_k = \frac{1}{m}\sum_{i=1}^{m} x_{ik}, \qquad \overline{x}_l = \frac{1}{m}\sum_{i=1}^{m} x_{il}.$$

Iš koreliacijos koeficientų, gautų naudojantis (2.1) formule, galima sudaryti *koreliacinę matricą* $R = \{r_{kl},\ k,l = 1,\ldots,n\}$. Matricos R įstrižainės elementai $r_{kk},\ k = 1,\ldots,n$, lygūs vienetui.

Parametrų x_k ir x_l *kovariacijos koeficientas* c_{kl} yra skaičiuojamas pagal šią formulę:

$$c_{kl} = \frac{1}{m-1}\sum_{i=1}^{m}(x_{ik} - \overline{x}_k)(x_{il} - \overline{x}_l). \tag{2.2}$$

Kai $k = l$, išraiška (2.2) yra dispersijos formulė, t. y. c_{kk} yra duomenų aibės parametro x_k dispersija.

Iš kovariacijos koeficientų, gautų remiantis (2.2) formule, galima suformuoti *kovariacinę matricą* $C = \{c_{kl},\ k,l = 1,\ldots,n\}$. Ši matrica yra simetrinė.

Iš (2.1) ir (2.2) formulių gauname, kad koreliacijos koeficientas

$$r_{kl} = \frac{c_{kl}}{\sqrt{c_{kk}c_{ll}}}.$$

Jei parametrai x_k ir x_l nekoreliuoti, tai jų kovariacijos koeficientas lygus 0: $c_{kl} = c_{lk} = 0,\ k \neq l$.

Apibrėžkime kovariacinės matricos tikrinius vektorius ir jų tikrines reikšmes. Matricos *tikrinis vektorius* (*eigenvector*) E_k ir jį atitinkanti *tikrinė*

reikšmė (*eigenvalue*) λ_k yra lygties $CE_k = \lambda_k E_k$ sprendinys. Šioje lygtyje E_k yra vektorius stulpelis, C – kovariacinė matrica, λ_k reikšmė randama iš charakteristinės lygties $|C - \lambda_k I| = 0$. Čia I yra vienetinė matrica, kurios matmenys tokie pat kaip matricos C; ženklu |.| apibrėžtas determinantas. Tikrinių vektorių skaičius yra lygūs duomenų matricą X sudarančių vektorių komponenčių skaičiui n. Rasti tikrinius vektorius ir tikrines reikšmes nėra trivialus uždavinys, tačiau yra sukurta nemažai šio uždavinio sprendimo metodų [97].

Surūšiuokime tikrinius vektorius E_k juos atitinkančių tikrinių reikšmių mažėjimo tvarka ($\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \ldots \geq \lambda_n$). Matrica $A = (E_1, E_2, \ldots, E_n)$ yra vadinama pagrindinių komponenčių matrica. Jos stulpeliai yra tikriniai vektoriai E_k, $k = 1, \ldots, n$, atitinkantys tikrines reikšmes $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \ldots \geq \lambda_n$. Kiekvienas matricos A vektorius stulpelis yra ortogonalus bet kuriam kitam.

Transformuokime duomenų vektorius X_i, $i = 1, \ldots, m$, pagal šią formulę:

$$Y_i = (X_i - \bar{X})A, \tag{2.3}$$

čia $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $\bar{X} = (\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_{in})$, $A = (E_1, E_2, \ldots, E_n)$.

Iš (2.3) formulės gauti $Y_i = (y_{i1}, y_{i2}, \ldots, y_{in})$ yra vektoriai (taškai) naujoje ortogonalioje koordinačių sistemoje $(y_1, y_2, \ldots, y_n)$, apibrėžtoje tikriniais vektoriais E_k, $k = 1, \ldots, n$. Vektorių Y_i, $i = 1, \ldots, m$, komponenčių $y_1, y_2, \ldots, y_n$ kovariacinė matrica

$$C_y = \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{pmatrix}.$$

Originalius duomenų vektorius X_i, $i = 1, \ldots, m$, galima išreikšti per vektorius Y_i pasinaudojus šia formule:

$$X_i = Y_i A^T + \bar{X}. \tag{2.4}$$

Ji gauta iš (2.3) formulės pritaikius ortogonalių matricų savybę, kad $A^{-1} = A^T$, čia A^{-1} yra atvirkštinė, A^T – transponuota matrica.

Daugiamačių duomenų transformacijai galima naudoti ne visus tikrinius vektorius, o tik pirmuosius. Tegul matrica A_d yra sudaryta iš d pirmųjų tikrinių vektorių. Tada galima sukurti transformaciją, analogišką (2.3) išraiškai:

$$Y_i = (X_i - \bar{X})A_d \,,\ i = 1,\ldots,m \,.$$

Tokiu būdu randama duomenų vektoriaus projekcija į d matmenų erdvę. Vizualizavimui pasirenkame $d = 2$ arba $d = 3$.

Tikrinių reikšmių $\lambda_1, \lambda_2, \ldots, \lambda_n$ savybės:

1) $\sum_{k=1}^{n} \lambda_k = \sum_{k=1}^{n} c_{kk}$;

2) $\lambda_1 = \max_k \lambda_k \geq \max_k c_{kk}$;

3) $\lambda_n = \min_k \lambda_k \leq \min_k c_{kk}$;

čia c_{kk} yra duomenų aibės parametro x_k dispersija, o tikrinė reikšmė λ_k yra transformuotų duomenų parametro y_k dispersija.

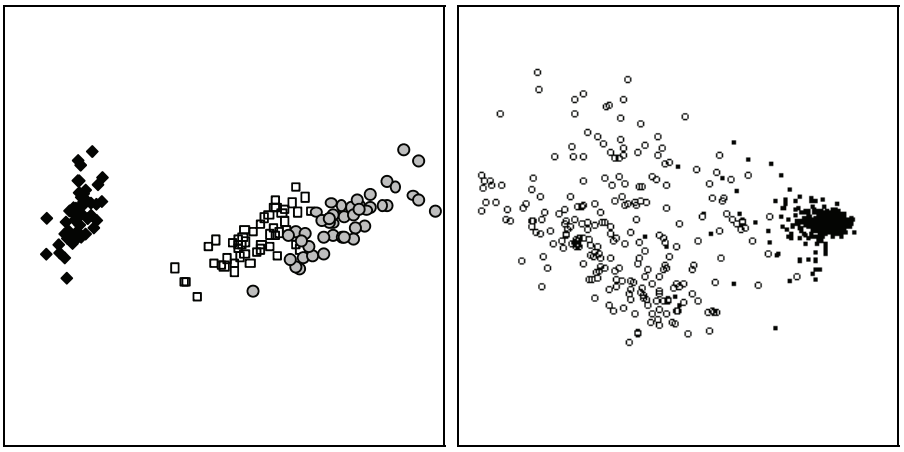
Iš antros savybės seka, kad pirmasis tikrinis vektorius E_1 aprašo naują parametrą y_1, kurio dispersija λ_1 yra pati didžiausia tarp $\lambda_1, \lambda_2, \ldots, \lambda_n$, t. y. šis parametras ir yra pirmoji pagrindinė komponentė. Antrasis tikrinis vektorius E_2 aprašo parametrą y_2, kurio dispersija yra antra pagal dydį ir kuris yra antroji pagrindinė komponentė, ir t. t.

Iš trečios savybės seka, kad paskutinis tikrinis vektorius E_n aprašo parametrą y_n, kurio dispersija bus pati mažiausia. Todėl naudojant tik d pirmųjų tikrinių vektorių, bus atsisakyta tos dalies po transformacijos gautų naujų parametrų, kurių dispersijos yra mažiausios.

Pagrindinėms komponentėms nustatyti užtenka rasti d didžiausių matricos C tikrinių reikšmių ir jas atitinkančių tikrinių vektorių. Matrica C turi išskirtines savybes – yra simetrinė ir neneigiamai apibrėžta, todėl taikomi specialūs greitai veikiantys algoritmai. Kita vertus, pagrindinėms komponentėms rasti naudojami ir dirbtiniai neuroniniai tinklai [68].

PKA metodo taikymo pavyzdžiai pateikti 2.27 paveiksle, kur irisų ir krūties vėžio duomenys atvaizduojami plokštumoje. Iš 2.27a paveikslo matome, kad pirmos veislės irisai (pažymėti juodai) „nutolę“ nuo kitų dviejų veislių, ryškios ribos tarp kitų dviejų veislių nėra. Didžioji dalis taškų 2.27b paveiksle, atitinkančių nepiktybinio naviko duomenis (juodi taškai), susikoncentruoja vienoje srityje, o kiti taškai, atitinkantys piktybinio naviko duomenis, pasiskirsto plačiai. Abiem atvejais stebime aiškiai išskiriančias taškų sankaupos sritis.

Pagrindinių komponenčių metodo privalumas yra jo idėjos paprastumas. Tai didžia dalimi nulėmė jo populiarumą ir platų taikymą.



a) b)

2.27 pav. PKA metodu vizualizuoti duomenys: a) irisų, b) krūties vėžio

Jeigu PKA metodas naudojamas ne duomenims vizualizuoti, kyla klausimas, kiek pagrindinių komponenčių *d* reikia parinkti. Dauguma skaičiaus *d* parinkimo taisyklių yra neformalios [152]. Galima parinkti tokį mažiausią skaičių *d*, kad procentinis dydis nuo visos dispersijos viršytų tam tikrą pasirinktą lygį, pavyzdžiui, 90%. Taigi išdėsčius tikrinius vektorius jų tikrinių reikšmių mažėjimo tvarka ir imant *d* pirmųjų vektorių, galima apskaičiuoti, kokia paklaida bus daroma. Pagal leidžiamą paklaidą imama daugiau ar mažiau pagrindinių komponenčių.

Kai kurie autoriai pagrindinių komponenčių analizėje, užuot naudojęsi kovariacine matrica, renkasi koreliacinę matricą. Tai yra ekvivalentu tam tikram parametrų $x_1, x_2, \ldots, x_n$ normavimui, o tik po to naujų parametrų (pagrindinių komponenčių) $y_1, y_2, \ldots, y_d$ paieškai.

Jei egzistuoja tiesinės priklausomybės tarp parametrų $x_1, x_2, \ldots, x_n$, tai, taikant pagrindinių komponenčių metodą, duomenų matmenų skaičius mažinamas su nedidelėmis paklaidomis. Tačiau bendru atveju gali egzistuoti netiesinės priklausomybės, kurių PKA metodas negali įvertinti, ir tai yra šio metodo trūkumas.

Vizualizavimo rezultatai, gauti taikant PKA metodą, labai priklauso nuo taškų atsiskyrėlių, nes šie taškai *n*-matėje erdvėje yra ryškiai nutolę nuo kitų, ir tai daro didelę įtaką kovariacinės matricos *C* elementų reikšmėms, kartu ir nustatytoms pagrindinėms komponentėms bei duomenų projekcijos kokybei visumoje.

Tiesinė diskriminantinė analizė

Skirtingai nuo daugelio kitų projekcijos metodų, tiesinės diskriminantinės analizės (TDA, *linear discriminant analysis*, LDA), vadinamasis Fišerio diskriminantinės analizės metodas realizuoja mokymo su mokytoju (*supervised*) strategiją. Mokymo su mokytoju metodais analizuojami duomenys, kurių savybės (pavyzdžiui, priklausymas kokiai nors klasei) yra žinomos ir jomis tiesiogiai yra pasinaudojama. TDA metodas transformuoja daugiamatės erdvės duomenis į mažesnio skaičiaus matmenų erdvę taip, kad klasių atskiriamumo kriterijaus reikšmė būtų optimali [33], [34], [55].

Tegul turime analizuojamų duomenų matricą

$$X = \{X_1, X_2, \ldots, X_m\} = \{x_{ij},\ i = 1, \ldots, m,\ \ j = 1, \ldots, n\},$$

kurios *i*-oje eilutėje yra vektorius $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$. Šios matricos X stulpeliai žymi duomenis apibūdinančius parametrus $x_1, x_2, \ldots, x_n$.

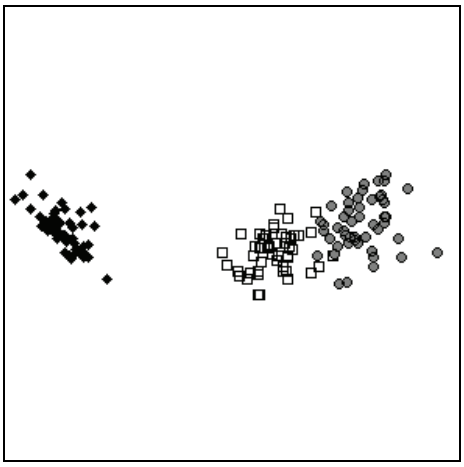
Tegul duomenų matrica X sudaryta iš kelių matricų $X^{(1)}, X^{(2)}, \ldots, X^{(k)}$, kurių eilutės yra vektoriai $X_i^{(j)}$, $i = 1, \ldots, m_j$, $j \in \{1, \ldots, k\}$, atitinkantys objektus, priklausančius vienai iš k klasių, čia m_j yra objektų, priklausančių *j*-ajai klasei, skaičius. Matrica $X^{(j)} = \{X_1^{(j)}, X_2^{(j)}, \ldots, X_{m_j}^{(j)}\}$, $j \in \{1, \ldots, k\}$, čia $X_i^{(j)} = (x_{i1}, x_{i2}, \ldots, x_{in})$.

TDA algoritmo schema:

1. Apskaičiuojamas analizuojamus objektus apibūdinančių parametrų visos aibės X reikšmių vidurkių vektorius $\overline{X} = (\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_n)$ ir kiekvienos klasės aibės $X^{(j)}$ reikšmių vidurkių vektoriai $\overline{X}^{(j)}$.
2. Pagal (2.2) formulę apskaičiuojami kovariacijos koeficientai, suformuojama bendra kovariacinė matrica C ir atskirų klasių kovariacinės matricos $C^{(j)}$, $j = 1, \ldots, k$.
3. Apskaičiuojamas išsibarstymas klasių viduje (*within-class scatter*) $S_w = \sum_{j=1}^{k} p_j C^{(j)}$, čia p_j yra apriorinė klasių tikimybė ($p_j = m_j / m$, čia m_j yra objektų, priklausančių *j*-ajai klasei, skaičius, m – visų analizuojamų objektų skaičius).
4. Apskaičiuojamas tarpklasinis išsibarstymas (*between-class scatter*) $S_b = C - S_w$.

5. Randami matricos $S_w^{-1}S_b$ tikriniai vektoriai ir tikrinės reikšmės. Jie surūšiuojami tikrinių reikšmių mažėjimo tvarka. Išrenkami d didžiausias tikrines reikšmes atitinkantys tikriniai vektoriai (būtinai turi būti $d < k$).
6. Apskaičiuojama duomenų transformacija $Y_i = (X_i - \bar{X})A_d$, $i = 1,...,m$, čia A_d yra matrica, sudaryta iš tikrinių vektorių, atitinkančių d didžiausių tikrinių reikšmių kaip vektorių stulpelių.

Tiesinės diskriminantinės analizės privalumas palyginti su pagrindinių komponenčių analize, yra tai, kad čia atsižvelgiama į duomenų klases. TDA metodu gautas daugiamačių taškų išsidėstymas dvimatėje plokštumoje vizualizuojant irisų duomenis parodytas 2.28 paveiksle.



2.28 pav. TDA metodu vizualizuoti irisų duomenys

Projekcijos paieška

Projekcijos paieškos (*projection pursuit*) metodą pasiūlė ir eksperimentiškai tyrinėjo J. B. Kruskalas [94]. Terminą *projection pursuit* pradėjo vartoti J. H. Friedmanas ir J. W. Tukey [53].

Projekcijos paieškos metodo, kaip ir daugelio kitų projekcijų metodų, tikslas yra rasti tokias tiesines komponenčių kombinacijas (dažniausiai dvimates ar trimates), kad transformuoti duomenys išlaikytų pradinių duomenų struktūrą [116].

Tarkime, turime duomenų matricą X, kurios eilutėse yra n-mačiai vektoriai. Erdvės R^n taškų projekciją į erdvę R^d galima išreikšti taip: $Y = XA$, čia $X = \{X_1, X_2, \ldots, X_m\} = \{x_{ij},\ i = 1, \ldots, m, j = 1, \ldots, n\}$, A yra matrica, sudaryta iš n eilučių ir d stulpelių, $Y = \{Y_1, Y_2, \ldots, Y_m\} = \{y_{ij}, i = 1, \ldots, m,\ j = 1, \ldots, d\}$ yra duomenų, gautų po projekcijos, matrica. Kyla klausimas, kaip parinkti matricą A?

Visų pirma reikia nuspręsti, kurią duomenų savybę norime išryškinti vizualizavimo metu. Tada reikia apsibrėžti matą, atspindintį šią savybę. Pavadinkime šį matą $I(Y)$. Dažnai jis vadinamas *indekso funkcija* (*index function*). Tarkime, norime išryškinti duomenų klasterius. Tuo atveju galėtų būti naudojamas statistikinis klasterizavimo matas – vidutinis atstumas iki artimiausio kaimyno (*mean nearest neighbour distance*). Kuo mažesnė šio mato reikšmė viename klasteryje, tuo duomenys yra geriau suklasterizuoti.

Tegul $I(Y)$ yra vidutinis atstumas iki artimiausio kaimyno. Išraiška $I(Y)$ gali būti pakeista į $I(XA)$. Projekcijos parinkimo uždavinys susiveda į optimizavimo uždavinį: reikia parinkti tokią matricą A, kad funkcijos $I(XA)$ reikšmė būtų mažiausia. Iš esmės tai ir yra projekcijos paieškos procesas [15].

Jeigu mus domina taškų atsiskyrėlių egzistavimas, tada uždavinys pakeičiamas taip, kad būtų ieškoma didžiausio vidutinio atstumo iki artimiausio kaimyno. Yra naudojamos ir kitos sudėtingesnės indeksų funkcijos [116].

2.2.2. Netiesinės projekcijos metodai

Daugiamatės skalės

Daugiamačių skalių (DS, *multidimensional scaling*, MDS) metodas [14] – tai grupė metodų, plačiai naudojamų daugiamačių duomenų analizei įvairiose šakose, ypač ekonomikoje, socialiniuose moksluose ir kt. Gausu šio metodo realizacijų, kurios skiriasi naudojamais vizualizavimo kokybės kriterijais, optimizavimo algoritmais ar prielaidomis apie duomenis.

Naudojantis DS, ieškoma daugiamačių duomenų projekcijų mažesnio skaičiaus matmenų erdvėje (dažniausiai R^2 arba R^3), siekiant išlaikyti analizuojamos aibės objektų artimumus – panašumus arba skirtingumus. Gautuose vaizduose panašūs objektai išdėstomi arčiau vieni kitų, o skirtingi – toliau vieni nuo kitų.

Pradiniai daugiamačių skalių metodo duomenys yra kvadratinė simetrinė matrica, kurios elementai nusako artimumą tarp analizuojamų objektų. Tai gali

būti arba panašumų, arba skirtingumų matrica. Paprasčiausiu atveju tai yra Euklido atstumų tarp objektų matrica. Tačiau bendruoju atveju, tai nebūtinai turi būti atstumai griežtai matematine prasme. Šiame skyrelyje skirtingumų matrica yra Euklido atstumų matrica, bendresnis skirtingumo matricų atvejis detaliai nagrinėjamas 3 skyriuje.

Vienas daugiamačių skalių metodų tikslų yra rasti optimalų daugiamačius objektus atitinkančių taškų (vektorių) vaizdą mažo skaičiaus matmenų erdvėje.

Tarkime, kiekvieną n-matį vektorių $X_i \in R^n$, $i \in \{1,\ldots,m\}$, atitinka mažesnio skaičiaus matmenų vektorius $Y_i \in R^d$, $d < n$. Atstumą tarp vektorių X_i ir X_j pažymėkime $d(X_i, X_j)$, o atstumą tarp vektorių Y_i ir Y_j – $d(Y_i, Y_j)$, $i, j = 1,\ldots,m$.

Naudojantis DS algoritmu, bandoma atstumus $d(Y_i, Y_j)$ priartinti prie atstumų $d(X_i, X_j)$. Jei naudojama kvadratinė paklaidos funkcija, tai minimizuojama tikslo funkcija E_{DS} gali būti užrašyta taip:

$$E_{DS} = \sum_{i<j} w_{ij} \left(d(X_i, X_j) - d(Y_i, Y_j) \right)^2 . \tag{2.5}$$

Paklaidos funkcija E_{DS} dar vadinama *Stress* funkcija. Dažnai naudojami tokie svoriai w_{ij}:

$$w_{ij} = \frac{1}{\sum_{i<j} \left(d(X_i, X_j) \right)^2},$$

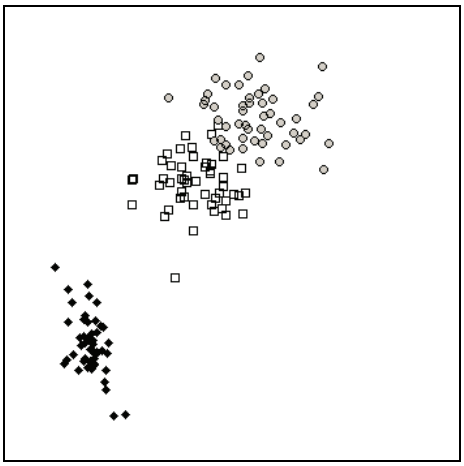
$$w_{ij} = \frac{1}{d(X_i, X_j) \sum_{k<l} d(X_k, X_l)},$$

$$w_{ij} = \frac{1}{m d(X_i, X_j)},$$

čia $d(X_i, X_j) \neq 0$, t. y. tarp vektorių $X_1, X_2, \ldots, X_m$ nėra sutampančių.

DS metodu gautas daugiamačių taškų išsidėstymas dvimatėje plokštumoje vizualizuojant irisų duomenis parodytas 2.29 paveiksle. Matome, kad taškai, atitinkantys pirmos klasės irisus (juodi), sudaro atskirą grupę. Ryškios ribos tarp kitų dviejų grupių nėra.

Detaliau daugiamačių skalių metodas nagrinėjamas 3 skyriuje.



2.29 pav. DS metodu vizualizuoti irisų duomenys

Sammono projekcija

Sammono projekcija, dažnai vadinama *Sammono metodu* ar *algoritmu*, yra netiesinis objektų, apibūdinamų daugeliu parametrų, atvaizdavimas mažesnio skaičiaus matmenų erdvėje R^d, dažniausiai $d = 2$ [128]. Jis priskiriamas prie daugiamačių skalių (DS) metodų grupės. Metodo idėja – atvaizduoti daugiamačius objektus atitinkančius vektorius mažesnio skaičiaus matmenų erdvėje išlaikant panašius atstumus tarp vektorių. Sammono projekcija minimizuoja projekcijos iškraipymą (paklaidą) E_S:

$$E_S = \frac{1}{\sum_{i<j} d(X_i, X_j)} \sum_{i<j} \frac{\left(d(X_i, X_j) - d(Y_i, Y_j)\right)^2}{d(X_i, X_j)}. \tag{2.6}$$

Funkcija E_S sutampa su E_{DS} funkcija, apibrėžta (2.5) formule, kai

$$w_{ij} = \frac{1}{d(X_i, X_j) \sum_{k<l} d(X_k, X_l)}.$$

Sammono paklaida (*Sammon's stress* (*error*)) E_S – tai matas, kuris parodo, kaip tiksliai išlaikomi atstumai tarp vektorių pereinant iš didesnio skaičiaus matmenų erdvės į mažesnio skaičiaus matmenų erdvę. Pagrindinis uždavinys – minimizuoti šią paklaidos funkciją E_S. Tam gali būti naudojami įvairūs optimizavimo metodai, nurodyti 3 skyriuje aprašant daugiamačių skalių

metodą. J. W. Sammonas [128] darbe pasiūlė vieną funkcijos E_S minimizavimo strategiją, kurią kai kurie tyrėjai vadina pseudo-Niutono, kiti – greičiausio nusileidimo (*steepest-descent*) minimizavimu [87].

J. W. Sammono pasiūlytu metodu dvimačių vektorių $Y_i \in R^2$ komponentės y_{ik}, $i = 1,\ldots,m$, $k = 1, 2$, randamos pagal iteracinę formulę

$$y_{ik}(t+1) = y_{ik}(t) - \eta \frac{\dfrac{\partial E_S(t)}{\partial y_{ik}(t)}}{\left|\dfrac{\partial^2 E_S(t)}{\partial y_{ik}^2(t)}\right|}, \tag{2.7}$$

čia t yra iteracijos numeris, o η – optimizavimo žingsnio ilgį reguliuojantis parametras. Viena *iteracija* perskaičiuojamos visų m vektorių $Y_i \in R^2$, $i = 1,\ldots,m,$ komponentės.

Dalinėms išvestinėms rasti naudojamos šios formulės:

$$\frac{\partial E_S}{\partial y_{ik}} = -\frac{2}{c}\sum_{i\neq j}\left(\frac{d(X_i,X_j)-d(Y_i,Y_j)}{d(X_i,X_j)d(Y_i,Y_j)}\right)(y_{ik}-y_{jk}),$$

$$\frac{\partial^2 E_S}{\partial y_{ik}^2} = -\frac{2}{c}\sum_{i\neq j}\frac{1}{d(X_i,X_j)d(Y_i,Y_j)}\left[\Big(d(X_i,X_j)-d(Y_i,Y_j)\Big) - \frac{(y_{ik}-y_{jk})^2 d(X_i,X_j)}{d^2(Y_i,Y_j)}\right],$$

$$c = \sum_{i<j} d(X_i,X_j).$$

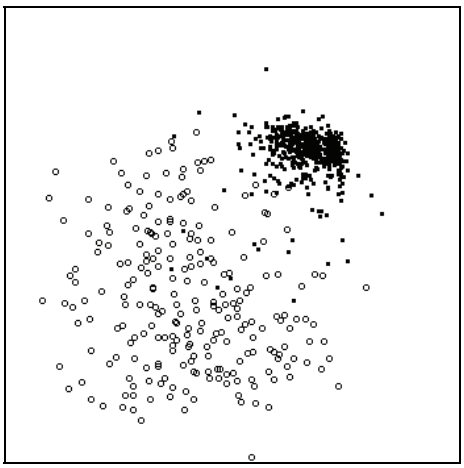
Gauta projekcijos paklaida E_S priklauso ir nuo parametro η ir nuo vektorių Y_i, $i = 1,\ldots,m$, komponenčių y_{ik}, $k = 1, 2$, pradinių reikšmių parinkimo. Eksperimentiškai nustatyta [87], [128], kad mažiausia paklaida gaunama, kai $\eta \in [0{,}3;\ 0{,}4]$, tačiau šio parametro reikšmė gali būti parinkta ir didesnė [44].

Nors šiuo metu yra ir kitų paklaidos E_S optimizavimo metodų, J. W. Sammono pasiūlytas variantas sėkmingai naudojamas daugelyje darbų [38], [39], [82], [87], [89].

Apibendrintoji Sammono algoritmo schema:

1. Skaičiuojami atstumai tarp analizuojamų vektorių pradinėje erdvėje.
2. Atsitiktinai parenkamos vaizdo erdvės vektorių komponenčių reikšmės.
3. Skaičiuojama projekcijos paklaida E_S.
4. Atnaujinamos vaizdo erdvės vektorių komponenčių reikšmės pagal (2.7) formulę.
5. Jeigu projekcijos paklaidos reikšmė mažesnė už pasirinktą slenkstį arba iteracijų skaičius viršija nustatytąjį, tuomet algoritmas sustabdomas, priešingu atveju grįžtama ir kartojama nuo 3 žingsnio.

Sammono metodu gautas daugiamačių taškų išsidėstymas dvimatėje plokštumoje vizualizuojant krūties vėžio duomenis parodytas 2.30 paveiksle.



2.30 pav. Sammono metodu vizualizuoti krūties vėžio duomenys

ISOMAP

ISOMAP metodą taip pat galima laikyti daugiamačių skalių (DS) grupės metodu, kuris skirtas daugiamačių duomenų matmenų skaičiui mažinti, tuo pačiu ir daugiamačiams duomenims vizualizuoti [137]. Šis metodas skiriasi nuo įprastinio DS metodo tuo, kad atstumų tarp analizuojamų objektų matas yra apibrėžiamas kitaip. Taikant ISOMAP metodą, daroma prielaida, kad pradinėje erdvėje analizuojamus duomenis atitinkantys taškai yra išsidėstę ant mažesnio skaičiaus matmenų netiesinės daugdaros, ir todėl skaičiavimuose naudojami geodeziniai atstumai.

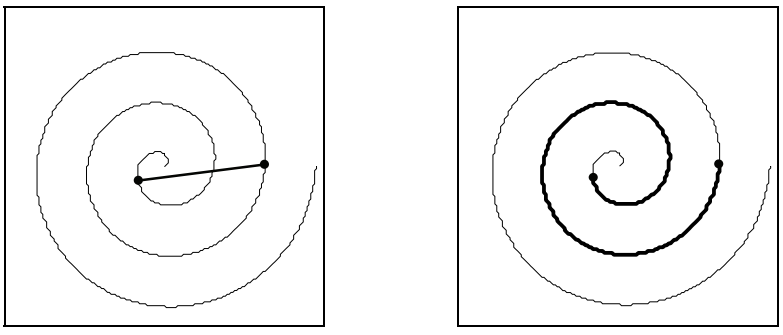
Pirmiausia apibrėžkime daugdaros sąvoką. *Daugdara* (*manifold*) – tai abstrakti topologinė matematinė erdvė, kurioje kiekvieno taško aplinka yra labai panaši į Euklido erdvę, tačiau šios erdvės globali struktūra daug sudėtingesnė.

Kalbant apie daugdarą svarbi yra matmens sąvoka šiame kontekste. Pavyzdžiui, linija yra vieno matmens, plokštuma – dviejų ir pan. Vienmatėje daugdaroje kiekvieno taško aplinka yra panaši į linijos segmentą. Linija, apskritimas, du atskiri apskritimai yra vienmatės daugdaros. Dvimatėje daugdaroje kiekvieno taško aplinka panaši į skritulį. Plokštuma, sferos paviršius, toro paviršius yra dvimatės daugdaros (*toras*, arba *toroidas*, geometrijoje yra sukimosi paviršius, kurį apibrėžia apskritimas, besisukantis aplink lygiagrečią su jo plokštuma ir jo neliečiančią ašį). Žemės paviršių taip pat galima laikyti dvimate daugdara.

Anksčiau nagrinėtuose metoduose atstumams tarp vektorių skaičiuoti paprastai yra naudojami Euklido atstumai. Skaičiuojant atstumus neatsižvelgiama į daugdaros formą. Tuomet susiduriama su sunkumais atvaizduojant netiesines duomenų struktūras, kaip kad spiralę ar pan.

ISOMAP metode artimumo tarp vizualizuojamų objektų (taškų) matas yra geodezinis atstumas. *Geodezinis*, arba *kreivinis*, *atstumas* – tai trumpiausio kelio ilgis einant daugdaros kreivu paviršiumi. Euklido atstumas tarp dviejų taškų parodytas 2.31a paveiksle, o 2.31b paveiksle pavaizduotas geodezinis atstumas tarp tų pačių taškų.

Norint apskaičiuoti geodezinius atstumus tarp duomenų taškų $X_1, X_2, \ldots, X_m$, pirmiausia reikia nustatyti kiekvieno taško X_i, $i \in \{1, \ldots, m\}$, artimiausius taškus, kuriuos vadinsime taškais kaimynais.



a) b)

2.31 pav. Atstumai tarp dviejų spiralės taškų: a) Euklido, b) geodezinis

Čia artimumo matas yra Euklido atstumas.Taškų kaimynų paieška gali būti organizuojama dvejopai: arba ieškoma nustatyto skaičiaus kaimynų, arba ieškoma kaimynų iš tam tikro fiksuoto dydžio spindulio hipersferos, kurios centras yra taškas X_i.

Tuomet sudaromas svorinis grafas, kuris jungia kiekvieną tašką X_i, $i \in \{1,\ldots,m\}$, su visais jo kaimynais. Grafo briaunų svoriai yra atstumai tarp taško X_i ir jo kaimynų. Naudojantis vienu iš trumpiausio kelio grafe radimo algoritmu, pavyzdžiui, Dijkstros [32], randami trumpiausių kelių ilgiai tarp visų taškų porų. Šie ilgiai ir yra geodezinių atstumų tarp taškų porų įverčiai.

Tokiu būdu suformuojama geodezinių atstumų matrica, kuri yra laikoma skirtingumo tarp analizuojamų objektų matrica. Šią matricą galima analizuoti bet kuriuo daugiamačių skalių metodu.

Taikant ISOMAP metodą ieškoma tokios transformacijos į vaizdo erdvę, kur būtų geriausiai išlaikomi geodeziniai atstumai tarp vizualizuojamų objektų (taškų).

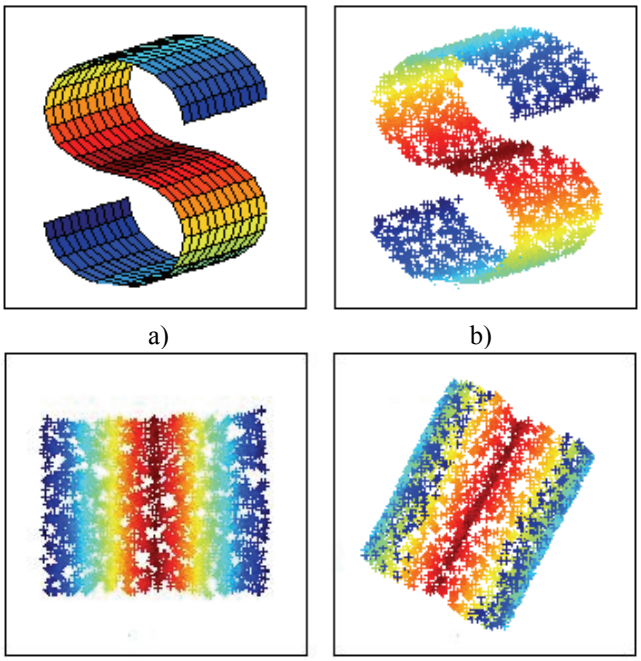
ISOMAP algoritmą sudaro trys etapai:

1. Daugiamatėje erdvėje randami kiekvieno taško kaimynai.
2. Skaičiuojami geodeziniai atstumai tarp visų taškų porų; suformuojama skirtingumų matrica.
3. Daugiamačių skalių metodu randamos daugiamačių taškų projekcijos mažesnio skaičiaus matmenų – vaizdo erdvėje.

S raidės daugdara pavaizduota 2.32a paveiksle, čia $n = 3$. Taškus, esančius ant šios daugdaros, matome 2.32b paveiksle ($m = 2000$).

Pateikta 2.32c paveiksle projekcija gauta ISOMAP metodu, o 2.32d paveiksle – gauta įprastiniu DS metodu, kai skirtingumų matrica yra Euklido atstumų matrica. Matome, kad ISOMAP metodu geriau išlaikoma daugdaros struktūra, S raidė tiesiog ištiesinama: tolimiausi taškai ant daugdaros (tamsiai mėlyna spalva) išlieka tolimiausi ir projekcijoje (žr. 2.32c pav.). Įprastiniu DS metodu gautoje projekcijoje tolimiausi taškai yra šviesiai mėlynos spalvos (žr. 2.32d pav.), tie taškai ant daugdaros yra tolimiausi Euklido atstumų prasme, bet ne geodezinių.

Kaip daugiamačių skalių procedūra realizuojant ISOMAP metodą [151] darbe buvo panaudota Sammono projekcija. Atstumus grafe koreguoti [31] darbe siūloma atsižvelgiant į duomenų struktūrą ir tankį. Kiti metodai, kuriuose skaičiuojami geodeziniai atstumai, – tai lokaliai tiesinis vaizdavimas ir kreivinių atstumų analizė.



2.32 pav. S raidės daugdaros pavyzdys: a) daugdara, b) taškai ant daugdaros, c) projekcija, gauta ISOMAP metodu, d) projekcija, gauta įprastiniu DS metodu

Lokaliai tiesinis vaizdavimas

2000 metais buvo pasiūlytas dar vienas daugiamačių duomenų matmenų skaičiaus mažinimo metodas – *lokaliai tiesinio vaizdavimo* (*locally linear embedding*, LLE) [125]. Šiuo metodu atvaizduojant daugiamačius duomenis į mažesnio skaičiaus matmenų erdvę, išlaikomi kaimynystės ryšiai tik tarp artimiausių taškų, bet atskleidžiama netiesinės daugdaros globali struktūra. Lokaliai tiesinio vaizdavimo algoritmo schema pateikta 2.33 paveiksle.

Algoritmą sudaro trys etapai:

- Nustatoma kiekvieno vizualizuojamo duomenų taško X_i, $i \in \{1,\ldots,m\}$, k artimiausių kaimynų. Artimumo matas yra Euklido atstumas. Taškų kaimynų paieška gali būti organizuojama dvejopai: arba ieškoma nustatyto skaičiaus kaimynų, arba ieškoma kaimynų iš tam tikro fiksuoto dydžio spindulio hipersferos, kurios centras yra taškas X_i.

- Kiekvienas daugiamatis taškas X_i išreiškiamas jo kaimynų tiesine kombinacija

$$X_i = \sum_{j=1}^{m} w_{ij} X_j,$$

čia svoriai $w_{ij} \neq 0$, jei taškas X_j yra taško X_i kaimynas, ir $w_{ij} = 0$, jei taškai X_i ir X_j nėra kaimynai. Svoriai w_{ij} turi būti parinkti taip, kad paklaida

$$E(W) = \sum_{i=1}^{m} \left(X_i - \sum_{j=1}^{m} w_{ij} X_j \right)^2$$

būtų minimali. Svoriai turi tenkinti sąlygą, kad visiems *i*

$$\sum_{j=1}^{m} w_{ij} = 1.$$

Tai yra tipinis mažiausiųjų kvadratų optimizavimo uždavinys su apribojimais. Jo sprendinys lengvai randamas sprendžiant tiesinių lygčių sistemą [125].

- Radus tinkamus svorius w_{ij}, projekcijos vektoriai Y_i, $i = 1,\ldots,m$, apskaičiuojami minimizuojant paklaidą

$$\Phi(Y) = \sum_{i=1}^{m} \left(Y_i - \sum_{j=1}^{m} w_{ij} Y_j \right)^2$$

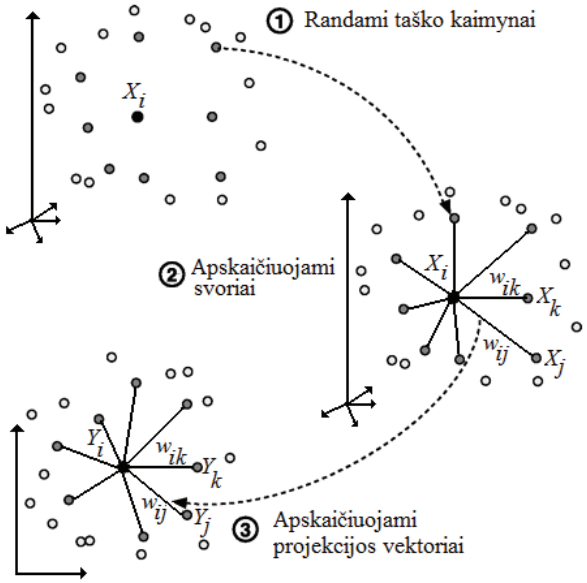
ir esant apribojimams

$$\sum_{i=1}^{m} Y_i = 0, \quad \frac{1}{m} \sum_{i=1}^{m} Y_i^T Y_i = I,$$

čia I – vienetinė matrica, sudaryta iš *d* eilučių ir *d* stulpelių, Y_i^T – transponuotas vektorius Y_i. Paprasčiausias būdas apskaičiuoti vektorių Y_i, $i = 1,\ldots,m$, *d*-mates komponentes yra rasti išretintos matricos

$$\bar{M} = (I - W)^T (I - W)$$

d tikrinių vektorių, atitinkančių mažiausias nenulines d tikrines reikšmes, čia I – vienetinė matrica, sudaryta iš m eilučių ir m stulpelių, $W = \{w_{ij},\ i, j = 1,\ldots,m\}$. Šie d tikriniai vektoriai ir yra vektoriai Y_i, $i = 1,\ldots,m$.



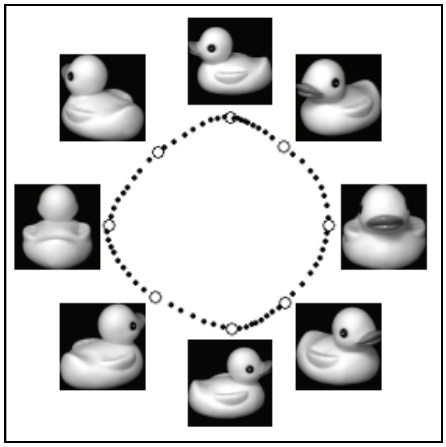
2.33 pav. Lokaliai tiesinio vaizdavimo algoritmo schema

Mažinant daugiamačių duomenų matmenų skaičių lokaliai tiesinio vaizdavimo metodu, pasiseka identifikuoti daugdaros nežinomą struktūrą. Tuo tarpu DS ir PKA metodais tolimi daugiamačiai taškai ant daugdaros atvaizduojami į artimus taškus plokštumoje, taigi suardoma daugdaros struktūra. DS grupės metodais stengiamasi išlaikyti santykinius atstumus tarp visų duomenų aibės taškų. Lokaliai tiesinio vaizdavimo metodas nereikalauja išlaikyti atstumų tarp labiausiai nutolusių duomenų taškų, o tik tarp artimiausių taškų kaimynų. Detaliau apie kaimynų skaičiaus parinkimą ir gauto vaizdo įvertinimą skaitykite [81] darbe.

Lokaliai tiesinio vaizdavimo metodas dažnai yra taikomas vizualizuoti daugiamačius vektorius, kurių komponenčių reikšmės atitinka paveikslėlių ar nuotraukų parametrus. Iš analizuojamo paveikslėlio ar nuotraukos taškų spalvinių savybių sudaromas vektorius, kurio matmenų skaičius yra labai

didelis. Analizuojamų duomenų aibė ypatinga tuo, kad ją sudaro vektoriai, atitinkantys kelis to paties objekto, pasukto tam tikrais kampais, paveikslėlius ar nuotraukas. Tuo atveju vektoriai vienas nuo kito nedaug skiriasi ir jie sudaro tam tikrą daugdarą. Lokaliai tiesinio vaizdavimo metodu vizualizuotus paveikslėlių duomenis, gautus laipsniškai sukant ančiuką, matome 2.34 paveiksle. Nespalvotos besisukančio ančiuko nuotraukos paimtos iš paveikslų bibliotekos http://www1.cs.columbia.edu/CAVE/software/softlib/coil-20.php.

Analizuojamų paveikslėlių, tuo pačiu vektorių skaičius $m = 72$. Kiekvienas paveikslėlis sudarytas iš 128x128 taškų, taigi vizualizuojamų vektorių matmenų skaičius $n = 16\,384$. Čia pateiktas daugiamačių taškų, atitinkančių pasukto ančiuko paveikslėlius, išsidėstymas dvimatėje plokštumoje. Didesni rutuliukai atitinka šalia pateiktus paveikslėlius. Ančiukas buvo laipsniškai sukamas aplink 360° kampu, todėl dvimačiai taškai išsidėstė ratu.



2.34 pav. Lokaliai tiesinio vaizdavimo metodu vizualizuoti besisukančio ančiuko paveikslėlių duomenys

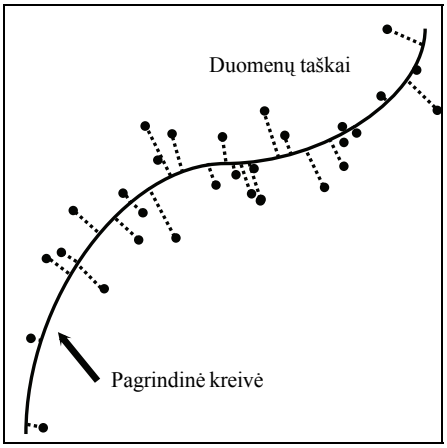
Kaip ir ISOMAP, lokaliai tiesinio vaizdavimo metodas yra labiau tinkamas vizualizuoti duomenis, kurie sudaro vientisą klasterį (grupę). Kai duomenų aibė sudaryta iš kelių visiškai atskirų duomenų klasterių, nėra gaunami geri vizualizavimo rezultatai. To priežastis yra tai, kad analizuojami duomenys nesudaro vientisos daugdaros.

Į lokaliai tiesinio vaizdavimo metodą panašūs yra *Laplaso tikriniai vaizdavimai* (*Laplacian eigenmaps*) [7], [8].

Pagrindinės kreivės

Pagrindinės kreivės (*principal curves*) yra pagrindinių komponenčių (*principal components*) apibendrinimas [26], [69]. Pagrindinė kreivė – tai glodžioji kreivė, brėžiama per duomenų centrinį tašką taip, kad vidutinis atstumas nuo duomenų taškų iki šios kreivės būtų minimalus, t. y. ši kreivė būtų kiek galima arčiau visų duomenų taškų (žr. 2.35 pav.).

Pagrindinių kreivių privalumai, palyginti su pagrindinėmis komponentėmis, tiesinės regresijos tiesėmis, glotniomis regresijos kreivėmis, aprašyti [69] darbe. Padaryta išvada, kad pagrindinė kreivė tiksliausiai aproksimuoja duomenų taškus.



2.35 pav. Pagrindinės kreivės pavyzdys

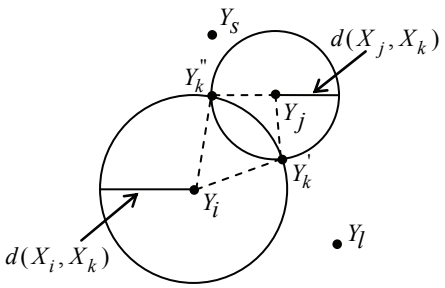
Trianguliacija

Daugiamačių taškų atvaizdavimo plokštumoje būdas taikant trianguliacijos metodą pateiktas [100] darbe. Daugiamačiai taškai atvaizduojami plokštumoje taip, kad atstumai tarp kiekvieno taško Y_i, kuris yra daugiamačio taško X_i dvimatė projekcija, iki kitų dviejų taškų Y_j ir Y_k, kurie yra taškų X_j ir X_k dvimatės projekcijos, būtų lygūs atstumams tarp daugiamačio taško X_i ir dviejų taškų X_j ir X_k. Atstumas tarp dvimačių taškų Y_j ir Y_k taip pat turi būti lygus atstumui tarp daugiamačių taškų X_j ir X_k. Taigi atstumai tarp dvimačių taškų $(2m-3)$ porų yra lygūs atstumams tarp juos atitinkančių

daugiamačių taškų, t. y. atstumai pereinant iš daugiamatės erdvės į dvimatę yra tiksliai išlaikomi, čia *m* – analizuojamų taškų skaičius.

Trianguliacijos metodo schema:

1. Nagrinėjami du daugiamačiai taškai X_i ir X_j. Plokštumoje yra atidedamos jų projekcijos Y_i ir Y_j taip, kad $d(X_i, X_j) = d(Y_i, Y_j)$.
2. Ieškoma taško X_k projekcijos Y_k plokštumoje, siekiant, kad atstumai $d(X_i, X_k) = d(Y_i, Y_k)$ ir $d(X_j, X_k) = d(Y_j, Y_k)$. Tam braižomi du apskritimai, kurių centrai yra Y_i ir Y_j, o spinduliai lygūs $d(X_i, X_k)$ ir $d(X_j, X_k)$ (žr. 2.36 pav.). Taško X_k projekcijos Y_k vieta bus ten, kur apskritimai susikerta arba liečiasi. Jei apskritimai tik liečiasi, tai bus vienintelis taškas Y_k. Tačiau apskritimai gali susikirsti dviejuose taškuose Y_k' ir Y_k''. Tada reikia nuspręsti, kurioje vietoje atvaizduoti tašką Y_k – ar $Y_k = Y_k'$, ar $Y_k = Y_k''$. Jei daugiamatis taškas X_k yra arčiau X_l, tai $Y_k = Y_k'$. Jei X_k yra arčiau X_s, tai $Y_k = Y_k''$.



2.36 pav. Daugiamačių taškų atvaizdavimas plokštumoje trianguliacijos metodu

Galimi du metodai, pagal kuriuos parenkama, tarp kurių trijų duomenų aibės taškų atstumų didumas turi būti tiksliai išlaikytas pereinant iš *n*-matės į dvimatę erdvę. Tai antrojo arčiausiojo kaimyno (*second nearest neighbour*) ir atramos taško (*reference point*) metodai. Prieš tai iš analizuojamų *n*-mačių taškų turi būti sudarytas minimalaus jungimo medis (*minimal spaning tree*).

Antrojo arčiausiojo kaimyno metodas. Sakykime, kad taškas X_j jau atvaizduotas plokštumoje ir gautas dvimatės jo projekcijos taškas Y_j. Taškas X_k yra tiesiogiai sujungtas su X_j minimalaus jungimo medyje, be to, iš visų

jau atvaizduotų taškų taškas X_j yra arčiausias taškui X_k. Tegul tarp visų atvaizduotų taškų, kurių atstumai iki X_j yra tiksliai išlaikyti pereinant iš *n*-matės erdvės į dvimatę, taškas X_i yra arčiausias iki taško X_k (taško X_i projekcija yra taškas Y_i). Įprastai X_i turi tiesioginę jungtį su X_j minimalaus jungimo medyje. Norime tašką X_k atvaizduoti plokštumoje, t. y. rasti dvimatį tašką Y_k. Taikant trianguliacijos metodą, taškas Y_k plokštumoje atidedamas taip, kad atstumai nuo jo iki dviejų taškų Y_i ir Y_j būtų lygūs atstumams nuo taško X_k iki dviejų taškų X_i ir X_j, kurie ir yra du artimiausi taško X_k kaimynai.

Atramos taško metodas. Jį taikant parenkamas vienas atramos taškas X_i, iki kurio atstumai nuo visų kitų taškų būtų visada išlaikomi pereinant iš *n*-matės į dvimatę erdvę. Taigi išlaikomi kiekvieno taško X_k atstumai iki dviejų taškų: iki to, kuris tiesiogiai sujungtas su tašku X_k minimalaus jungimo medyje, ir iki atramos taško X_i.

Darbe [100] pasiūlytos taisyklės, pagal kurias taškai surūšiuojami tam tikra tvarka ir tada iš eilės jie atvaizduojami plokštumoje.

Trianguliacijos metodo jungimas su Sammono projekcija. Darbe [13] pasiūlyta trianguliacijos metodą jungti su Sammono algoritmu. Trianguliacijos metodas yra pakankamai greitas, tačiau pereinant iš *n*-matės į dvimatę erdvę juo tiksliai išlaikoma tik $(2m-3)$ atstumų. Sammono algoritmu bandoma išlaikyti visus santykinius atstumus $m(m-1)/2$ tarp taškų, tačiau jis yra gana lėtas. Norint pagreitinti skaičiavimus, tačiau šiek tiek atsisakyti tikslumo, verta naudotis šių dviejų metodų junginiu. Iš pradžių parenkama $\overline{m}$ *n*-mačių taškų $X_1, X_2, \ldots, X_{\overline{m}}$, $\overline{m} < m$, ir pagal Sammono algoritmą atvaizduojama plokštumoje. Jie vadinami baziniais taškais. Likusieji $(m-\overline{m})$ taškų nuosekliai vienas po kito atvaizduojami plokštumoje taikant trianguliacijos metodą taip, kad būtų tiksliai išlaikyti atstumai nuo kiekvieno taško, nepriklausančio baziniams taškams, iki artimiausių dviejų bazinių taškų. Kyla klausimas, koks turi būti skaičius $\overline{m}$? Kai žinomos taškų klasės, baziniais taškais turi būti keli atstovai iš kiekvienos klasės. Kitais atvejais duomenys prieš analizuojant klasterizuojami ir jų centrai imami baziniais taškais. Darbe [13] tokio junginio rezultatai palyginti su kitais projekcijos metodais, parodyta, kad toks metodas veikia gana greitai, o jo tikslumas gana didelis. Taip pat šis metodas gali būti taikomas, kai reikia greitai atvaizduoti naujus analizuojamos aibės taškus.

3. Daugiamatės skalės

Daugiamatės skalės (DS) (*multidimensional scaling*, MDS), nagrinėjamos [14], [21], [25], [95], [108], [130] ir [139] darbuose, naudojamos daugiamačių duomenų struktūros analizei dvimatėje arba trimatėje erdvėje. Tai vienmačių skalių, kai objektai išdėstomi atkarpoje, apibendrinimas daugiamatėje erdvėje. Teoriškai objektų atvaizdavimą galima nagrinėti bet kokio skaičiaus matmenų erdvėje, tačiau praktiškai naudingos tik tokios, kurias galima vizualizuoti turimomis priemonėmis. Dvimatės skalės yra naudojamos dažniausiai, tačiau, plintant trimačio vaizdavimo priemonėms, trimatės skalės tampa vis labiau patrauklesnės.

Daugiamatės skalės taikomos daugelyje sričių – ir psichometrikos [133], ir rinkos analizės [30], [117], ir telekomunikacijų [65], ir farmakologijos [159].

Naudojant daugiamačių skalių metodą, ne tik daugiamačiai vektoriai, bet ir bendresnės daugiamačių duomenų aibės gali būti vizualizuojamos, nes pakankami duomenys šiam metodui yra objektų *artimumas*, apibrėžtas objektų porų *skirtingumu*. Objektų skirtingumas gali būti įvertinamas psichologiniuose tyrimuose. Tokių duomenų matmenų (parametrų) skaičius gali būti nežinomas, skirtingumas gali neturėti metrinių savybių, pavyzdžiui jam gali negalioti trikampio nelygybė.

Jei duomenys pateikiami daugiamačiais vektoriais, jų skirtingumas įvertinamas atstumu daugiamatėje *duomenų erdvėje*. DS metodo esmė – minimizuoti *įtempimo funkciją*, kuria naudojantis lyginami objektų skirtingumai su atstumais tarp tuos objektus atvaizduojančių taškų *vaizdo erdvėje*.

3.1. Daugiamačių skalių uždavinių formulavimas

Daugiamačių skalių metodu sprendžiamas uždavinys, kaip *m* objektų, apibrėžtų artimumo duomenimis, gali būti patikimai atvaizduoti taškais mažo skaičiaus matmenų vaizdo erdvėje. Objektų artimumas yra apibrėžiamas jų porų *skirtingumu* (nepanašumu), *i*-ojo ir *j*-ojo objektų skirtingumas nusakomas realiuoju skaičiumi $\delta_{ij}, i, j = 1,\ldots,m$. Taigi, *d*-matėje erdvėje ieškoma taškų $Y_i \in R^d, i = 1,\ldots,m$, tarp kurių atstumai atitiktų skirtingumus.

Atvaizdavimo kokybė matuojama *įtempimo funkcija*, kuria naudojantis lyginamas objektų skirtingumas su atstumu tarp juos atvaizduojančių taškų. Dažniausiai yra naudojami *Minkovskio atstumai*:

$$d_q(Y_i,Y_j)=\left(\sum_{k=1}^{d}\left|y_{ik}-y_{jk}\right|^q\right)^{\frac{1}{q}}, \tag{3.1}$$

čia $Y_i=(y_{i1},y_{i2},\ldots,y_{id})$ ir $Y_j=(y_{j1},y_{j2},\ldots,y_{jd})$. Kai $q=2$, pagal šią formulę apskaičiuojami *Euklido atstumai*. Kai $q=1$, gaunami *miesto kvartalo*, vadinamieji *Manheteno* ar *pirmos normos*, *atstumai*.

Nors DS dažniausiai naudojami Euklido atstumai, DS gali būti informatyvesnės, kai skaičiuojami kiti Minkovskio atstumai vaizdo erdvėje. Dažniausiai yra naudojama *mažiausiųjų kvadratų įtempimo funkcija* (dažnai literatūroje vadinama *Stress*)

$$S(Y)=\sum_{i=1}^{m}\sum_{j=1}^{m}w_{ij}\left(d(Y_i,Y_j)-\delta_{ij}\right)^2, \tag{3.2}$$

čia $Y=(Y_1,Y_2,\ldots,Y_m)$, $d(Y_i,Y_j)$ žymi atstumą tarp taškų Y_i ir Y_j, $w_{ij}>0$ yra teigiami svoriai. Kadangi $d(Y_i,Y_j)=d(Y_j,Y_i)$, $d(Y_i,Y_i)=0$, o paprastai $\delta_{ij}=\delta_{ji}$, $\delta_{ii}=0$, $w_{ij}=w_{ji}$, dėl to ši funkcija gali būti apibrėžta glaustai, sumuojant tik dalį skirtumų:

$$\sigma_r(Y)=\sum_{i<j}w_{ij}\left(d(Y_i,Y_j)-\delta_{ij}\right)^2. \tag{3.3}$$

Normuotoji mažiausiųjų kvadratų įtempimo funkcija

$$\sigma_n(Y)=\frac{\sum_{i<j}w_{ij}\left(d(Y_i,Y_j)-\delta_{ij}\right)^2}{\sum_{i<j}w_{ij}\delta_{ij}^2}$$

parodo santykinį įtempimą ir gali būti naudojama skirtingų duomenų rezultatams palyginti.

Kiekybiniam vizualizavimo įvertinimui dažnai taikoma santykinė paklaida

$$E(Y)=\sqrt{\frac{\sum_{i<j}w_{ij}\left(d(Y_i,Y_j)-\delta_{ij}\right)^2}{\sum_{i<j}w_{ij}\delta_{ij}^2}}, \tag{3.4}$$

o ne įtempimo funkcijos reikšmė.

Taip sumažinama objektų skaičiaus ir duomenų erdvės atstumų tipo (pavyzdžiui, *q* reikšmės, kai naudojami Minkovskio atstumai) įtaka.

Visur diferencijuojama įtempimo funkcija (dažnai literatūroje vadinama *S-Stress*) apibrėžiama formule

$$S_S(Y) = \sum_{i<j} w_{ij} \left(d^2(Y_i, Y_j) - \delta_{ij}^2 \right)^2. \tag{3.5}$$

Ji patogesnė, nes yra paprasčiau minimizuoti dėl išvengtų diferencijavimo problemų. Tačiau dėl kvadratų (3.5) formulėje labiau akcentuojami didesni skirtingumai negu mažesni. Tai yra šios įtempimo funkcijos trūkumas.

Kartais vietoje mažiausiųjų kvadratų įtempimo funkcijos naudojama *mažiausiojo absoliučiojo nuokrypio* ($L1$ normos) funkcija

$$S_{L1}(Y) = \sum_{i<j} w_{ij} \left| d(Y_i, Y_j) - \delta_{ij} \right|. \tag{3.6}$$

Objektų vaizdas randamas minimizuojant įtempimo funkciją: *d*-matėje erdvėje ieškoma tokių *m* taškų koordinačių, kad įtempimo funkcija būtų minimali. Matematiškai optimizavimo uždavinys formuluojamas taip: reikia rasti netiesinės $m \times d$ tolydžiųjų kintamųjų funkcijos (vadinamosios tikslo funkcijos) $f(Y): R^{m \times d} \to R$ minimumą

$$f^* = \min f(Y),$$

ir globalaus minimumo tašką Y^*, kuriame funkcijos reikšmė lygi funkcijos minimumui:

$$Y^*: f(Y^*) = f^*.$$

Įtempimo funkciją minimizuoti gan sudėtinga, nes:

- yra daug lokalių minimumų taškų, o interpretuojant skirtingus lokalių minimumų taškus atitinkančius vaizdus gaunami skirtingi rezultatai, dėl to svarbu rasti globalų minimumą ir jį atitinkantį vaizdą;
- praktinių uždavinių minimizavimo kintamųjų skaičius $m \times d$ paprastai yra didelis;
- įtempimo funkcija yra ne visur diferencijuojama;
- įtempimo funkcija yra invariantinė perkėlimo, sukimo ir atspindžių atžvilgiu, t. y. analizuojant pateiktus duomenis vizualiai nėra svarbu, ar jų visuma bus paslinkta, ar pasukta, ar pateiktas jos atspindys kokios nors ašies atžvilgiu.

Invariantiškumo išvengti galima įvedus papildomus apribojimus arba fiksavus dalį vaizdo taškų koordinačių. Pavyzdžiui, invariantiškumo perkėlimo atžvilgiu galima išvengti centruojant vaizdą, nustačius tam tikrus minimizavimo apribojimus, t. y. pareikalavus, kad vaizdo taškų koordinačių reikšmių sumos būtų lygios nuliui:

$$\sum_{i=1}^{m} y_{ik} = 0,\ k = 1,\ldots,d, \tag{3.7}$$

arba koordinačių pradžioje fiksavus pirmąjį objektą vaizduojantį tašką

$$y_{1k} = 0,\ k = 1,\ldots,d.$$

Invariantiškumo atspindžių ir sukimo atžvilgiu, esančio naudojantis Euklido atstumais, galima išvengti pareikalavus, kad antrąjį objektą vaizduojantis taškas būtų tik teigiamoje pirmosios koordinatės ašyje. Siekiant išvengti invariantiškumo atspindžių ir sukimo aplink pirmąją koordinačių ašį atžvilgiu, galima leisti trečiajam taškui būti tik teigiamoje pusplokštumėje – jo antroji koordinatė turi būti teigiama. Jeigu vaizdo erdvė yra didesnio skaičiaus matmenų negu dvimatė, antrojo taško trečioji koordinatė turi būti lygi nuliui, o siekiant išvengti invariantiškumo atspindžių atveju, trečiojo taško trečioji koordinatė turi būti teigiama. Tai galima aprašyti apribojimais:

$$y_{ik} = 0,\ i = 2,\ldots,d,\ k = i,\ldots,d,$$

$$y_{i(i-1)} \geq 0,\ i = 2,\ldots,d+1.$$

Tokiu atveju tik nefiksuotos koordinatės yra minimizavimo uždavinio kintamieji, taigi jų skaičius yra šiek tiek sumažinamas.

3.2. Įtempimo funkcijos diferencijuojamumo savybės

Kaip jau buvo minėta, DS metodo esmė yra įtempimo funkcijos minimizavimas. Daug yra DS algoritmų, kuriuose minimizuojant įtempimo funkciją naudojami pagalbiniai lokalaus minimizavimo algoritmai. Juos renkantis, labai svarbu atsižvelgti į tikslo funkcijos diferencijuojamumą minimumo taške.

Mažiausiųjų kvadratų įtempimo funkcijos (3.2) su Euklido atstumais diferencijuojamumo lokalaus minimumo taške analizė, pristatyta [23] darbe, [64] darbe apibendrinta Minkovskio atstumams. Kai $w_{ij}\delta_{ij} > 0$ visiems $i, j = 1,\ldots,m, i \neq j,$ mažiausiųjų kvadratų įtempimo funkcijos lokalaus

minimumo tašką atitinkančiame DS vaizde skirtingus objektus vaizduojantys taškai nesutampa – atstumai tarp vaizdo taškų yra teigiami. Mažiausiųjų kvadratų įtempimo funkcija su (3.1) Minkovskio atstumais, kai $q > 1$, yra nediferencijuojama tik tada, kai atstumas tarp dviejų vaizdo taškų yra lygus nuliui.

Teigiami atstumai parodo, kad įtempimo funkcija su Minkovskio atstumais $q > 1$ yra diferencijuojama lokalaus minimumo taške. Taigi tokiu atveju galima naudotis gradientiniais lokalios paieškos metodais, tik negalima jų pradėti nuo optimizavimo kintamųjų reikšmių, kurios atitinka sutampančius bent dviejų objektų vaizdo taškus, nes tokiu atveju įtempimo funkcija yra nediferencijuojama. Tas pats galioja ir $d = 1$ atveju, kai visi Minkovskio atstumai yra tokie patys nepriklausomai nuo q.

Tačiau mažiausiųjų kvadratų įtempimo funkcija su miesto kvartalo atstumais ($q = 1$) yra nediferencijuojama ir tada, kai dviejų vaizdo taškų bent viena koordinatė sutampa. Dėl to teigiami atstumai yra nepakankama mažiausiųjų kvadratų įtempimo funkcijos su miesto kvartalo atstumais diferencijuojamumo sąlyga.

Kaip buvo parodyta [157] darbe, lokalaus minimumo taškus atitinkančiuose DS vaizduose skirtingus objektus vaizduojančių taškų koordinatės gali sutapti, kai $d > 1$. Dėl to mažiausiųjų kvadratų įtempimo funkcija su miesto kvartalo atstumais gali būti nediferencijuojama lokalaus minimumo taškuose.

Detaliau panagrinėkime diferencijuojamumą mažiausiųjų kvadratų įtempimo funkcijos $S(Y)$, apibrėžtos (3.2) formule, kurioje $d(Y_i, Y_j)$ yra miesto kvartalo atstumai. Tarkime, Y^* yra funkcijos $S(Y)$ lokalaus minimumo taškas. Tada kryptinė išvestinė pagal laisvai pasirinktą (vienetinį) krypties vektorių V yra neneigiama: $D_V S(Y^*) \geq 0$. Dėl to laisvai pasirinktam vektoriui V galioja nelygybė

$$D_V S(Y^*) + D_{-V} S(Y^*) \geq 0. \tag{3.8}$$

Funkcijos $S(Y)$ kryptinės išvestinės pagal krypties vektorių V taške Y^* išraiškoje

$$D_V S(Y^*) = \sum_{i=1}^{m} \sum_{j=1}^{m} 2w_{ij} \left(d(Y_i^*, Y_j^*) - \delta_{ij} \right) D_V d(Y_i^*, Y_j^*) \tag{3.9}$$

yra $D_V d(Y_i^*, Y_j^*)$, kuri gali būti randama naudojantis formule

$$D_{V_{ijk}} d(Y_i^*, Y_j^*) = \begin{cases} v_{ik} - v_{jk}, \text{ jei } y_{ik}^* - y_{jk}^* > 0, \\ v_{jk} - v_{ik}, \text{ jei } y_{ik}^* - y_{jk}^* < 0, \\ \left| v_{ik} - v_{jk} \right|, \text{ jei } y_{ik}^* - y_{jk}^* = 0, \end{cases}$$

čia V_{ijk} žymi vektorių, kurio visi elementai yra lygūs nuliui, išskyrus atitinkančius $y_{ik}^*, y_{jk}^*, k = 1, \ldots, d$. Šią formulę galima užrašyti trumpiau:

$$D_{V_{ijk}} d(Y_i^*, Y_j^*) = \left| v_{ik} - v_{jk} \right| \operatorname{sign}\left((y_{ik}^* - y_{jk}^*)(v_{ik} - v_{jk}) \right), \tag{3.10}$$

čia sign(·) žymi nesimetrinę *signum* funkciją: $\operatorname{sign}(t) = 1$, kai $t \geq 0$, ir $\operatorname{sign}(t) = -1$, kai $t < 0$. Įvertinę (3.9) ir (3.10) formules, gauname

$$D_V d(Y_i^*, Y_j^*) = \sum_{k=1}^{d} \left| v_{ik} - v_{jk} \right| \operatorname{sign}\left((y_{ik}^* - y_{jk}^*)(v_{ik} - v_{jk}) \right). \tag{3.11}$$

Iš (3.8), (3.9) ir (3.11) formulių išplaukia, kad

$$4 \sum_{k=1}^{d} \sum_{i,j \in Q_k} w_{ij} \left(d(Y_i^*, Y_j^*) - \delta_{ij} \right) \left| v_{ik} - v_{jk} \right| \geq 0, \tag{3.12}$$

čia $Q_k = \{(i, j) : y_{ik}^* = y_{jk}^*\}$.

Kai

$$d(Y_i^*, Y_j^*) = 0, \; d(V_i, V_j) > 0 \;\text{ ir }\; d(V_k, V_l) = 0, \;\; k, l \neq i, j,$$

(3.12) nelygybė negalioja, todėl lokalaus minimumo taške Y^* nelygybė $d(Y_i^*, Y_j^*) > 0$ turi galioti visiems $i \neq j$.

Teigiami atstumai $d(Y_i^*, Y_j^*) > 0$ parodo, kad skirtingus objektus vaizduojantys taškai DS nesutampa, o mažiausiųjų kvadratų įtempimo funkcija su Minkovskio atstumais, kai $q > 1$, yra diferencijuojama lokalaus minimumo taške. Tačiau teigiami atstumai yra nepakankama mažiausiųjų kvadratų įtempimo funkcijos su miesto kvartalo atstumais ($q = 1$) diferencijuojamumo sąlyga, kai $d > 1$. Tačiau tai neįrodo, kad yra lokalaus minimumo taškų su

nediferencijuojama įtempimo funkcija. Įrodymui panagrinėkime paprastą dvimatės skalės ($d = 2$) uždavinio pavyzdį, kuriame įtempimo funkcija su miesto kvartalo atstumais yra nediferencijuojama lokalaus minimumo taške. Uždavinio duomenys – skirtingumai $\delta_{12} = \delta_{14} = \delta_{23} = \delta_{34} = 1$, $\delta_{13} = \delta_{24} = 3$:

$$\Delta = \begin{pmatrix} 0 & 1 & 3 & 1 \\ 1 & 0 & 1 & 3 \\ 3 & 1 & 0 & 1 \\ 1 & 3 & 1 & 0 \end{pmatrix},$$

o svoriai lygūs $w_{ij} = 1$.

Kvadrato, su centru koordinačių pradžioje ir kraštinės ilgiu 4/3, viršūnės yra potencialus tokių duomenų vaizdas DS. Toks vaizdas atitiktų minimizavimo uždavinio kintamųjų vektoriaus reikšmes

$$y_{11}^* = y_{21}^* = y_{12}^* = y_{42}^* = -\frac{2}{3}, \quad y_{31}^* = y_{41}^* = y_{22}^* = y_{32}^* = \frac{2}{3}.$$

Parodysime, kad toks Y^* yra funkcijos $S(Y)$ lokalaus minimumo taškas, kuriame įtempimo funkcija yra nediferencijuojama. Funkcijos $S(Y)$ kryptinė išvestinė taške Y^* pagal laisvai pasirinktą (vienetinį) krypties vektorių V yra

$$D_V S(Y^*) = 4\left(|v_{11} - v_{21}| + |v_{12} - v_{42}| + |v_{22} - v_{32}| + |v_{31} - v_{41}|\right) / 3 \geq 0. \qquad (3.13)$$

Taigi Y^* yra lokalaus minimumo taškas, kuriame funkcija yra nediferencijuojama, nes kryptinė išvestinė yra teigiama, t. y.

$$D_V S(Y^*) > 0, \qquad (3.14)$$

nebent visi sumuojami (3.13) išraiškos elementai yra lygūs nuliui. Tokiu atveju krypties vektorius turi tenkinti lygybes

$$v_{11} = v_{21}, \quad v_{12} = v_{42}, \quad v_{22} = v_{32}, \quad v_{31} = v_{41}. \qquad (3.15)$$

Tada $S(Y^* + tV)$ yra diferencijuojama pagal t. Gana ilga pradinė

$$\frac{d^2}{dt^2} S(Y^* + tV)$$

išraiška algebrinėmis operacijomis gali būti suvedama į tokią:

$$\frac{d^2}{dt^2}S(Y^* + tV) = 4\Big((v_{12} - v_{22})^2 + (v_{11} - v_{31} + v_{12} - v_{32})^2 + \\ + (v_{11} - v_{41})^2 + (v_{21} - v_{31})^2 + \\ + (v_{21} - v_{41} + v_{22} - v_{42})^2 + (v_{32} - v_{42})\Big)^2. \tag{3.16}$$

Iš čia

$$\frac{d^2}{dt^2}S(Y^* + tV)\Big|_{t=0} > 0 \tag{3.17}$$

visiems (3.15) lygybes tenkinantiems vektoriams, nebent visi sumuojami (3.16) išraiškos elementai yra lygūs nuliui. Kai V tenkina sąlygą

$$\frac{d^2}{dt^2}S(Y^* + tV) = 0$$

ir (3.15) lygybes, tai turi būti ir

$$v_{11} = v_{21} = v_{31} = v_{41}, \quad v_{12} = v_{22} = v_{32} = v_{42}.$$

Tačiau $S(Y)$ yra invariantinė perkėlimo atžvilgiu, todėl

$$S(Y^* + tV) = S(Y^*).$$

Vadinasi, galima teigti, kad (3.14) ir (3.17) formulės įrodo, kad Y^* yra funkcijos $S(Y)$ lokalaus minimumo taškas, kuriame mažiausiųjų kvadratų įtempimo funkcija su miesto kvartalo atstumais yra nediferencijuojama. Dėl to greitai konverguojantys lokalaus nusileidimo metodai, tokie kaip įvairios Niutono metodo versijos, yra netinkami minimizuoti šią funkciją.

3.3. Minimizavimo algoritmai daugiamatėms skalėms

Vienas populiariausių minimizavimo algoritmų DS yra SMACOF (*Scaling by Majorizing a Complicated Function*) [22]. Jis yra pagrįstas tikslo funkcijos mažorizavimu [60]. Čia mažiausiųjų kvadratų įtempimo funkcijos *Stress*, apibrėžtos (3.2) ir (3.3) formulėmis, minimizavimas pakeistas iteraciniu pagalbinės funkcijos minimizavimu, kuris yra daug paprastesnis. DS algoritmų konvergavimas tirtas [24] darbe ir įrodyta, kad mažorizavimo metodas, skirtingai nuo standartinio Niutono metodo, yra globaliai konverguojantis. Tai yra pasiekia lokalaus minimumo tašką pradėjus algoritmą iš bet kur, o ne tik iš

jo aplinkos. Dažniausiai konvergavimas yra tiesinis, o konvergavimo greitis – artimas vienetui. Mažorizavimo algoritmas skirtas DS su Euklido atstumais, tačiau buvo apibendrintas Minkovskio atstumams, kai $1 \le q \le 2$. Iš dalies mažorizavimu pagrįstas algoritmas Minkovskio atstumams su kitokiu q buvo pasiūlytas [64] darbe.

Nuoseklaus įvertinimo (*sequential estimation*) sąvoka DS pradėta vartoti [115] darbe. Taikant nuoseklaus įvertinimo metodą, vaizdas atnaujinamas, kai į analizuojamą duomenų aibę įtraukiami nauji objektai. Metodas remiasi tuo, kad vieno objekto papildymas reikalauja daug mažiau skaičiavimų negu visos duomenų aibės vaizdo DS paieška nuo pradžių minimizuojant įtempimo funkciją.

Literatūroje dėmesys skiriamas ne tik dažniausiai naudojamai *Stress* įtempimo funkcijai. Globalizuotas Niutono metodas DS su *Stress* ir *S-Stress* įtempimo funkcijomis, apibrėžtoms (3.2) ir (3.5) formulėmis, išplėtotas [83] darbe. Deterministinis atkaitinimo modeliavimo algoritmas *S-Stress* įtempimo funkcijai yra pasiūlytas ir eksperimentiškai palygintas su gradientinio nusileidimo metodais [86] darbe.

Paprastai įtempimo funkcijos turi daug lokalių minimumų taškų, o interpretuojant skirtingus lokalių minimumų taškus atitinkančius vaizdus gaunami skirtingi rezultatai. Dėl to svarbu rasti globalųjį minimumą ir jį atitinkantį vaizdą. Vienas iš paprasčiausių būdų yra lokalios paieškos iš įvairių atsitiktinių pradžios taškų (literatūroje vadinamas *multistart*). Tokia paieška dažnai naudojama DS dėl to, kad yra paprasta, tačiau sudėtingesni globalaus optimizavimo algoritmai yra daug patikimesni.

Tuneliavimo (*tunneling*) globalaus minimizavimo metodas buvo pasiūlytas ir pritaikytas DS su Minkovskio atstumais [61] darbe. Taikant tuneliavimo metodą, lokalus minimizavimas yra derinamas su tuneliavimo žingsniu. Ieškoma kito vaizdo su ta pačia įtempimo funkcijos reikšme kaip prieš tai rastas lokalus minimumas. Paeiliui kartojant lokalios paieškos ir tuneliavimo žingsnius, randami geresni lokalūs minimumai, iš kurių paskutinis dažnai būna globalus.

DS metodas, kuriame lokalus minimizavimas derinamas su evoliucine paieška naujų pradinių taškų generavimui, pasiūlytas [109] darbe. Tokio algoritmo efektyvumas buvo tirtas eksperimentiškai. Rezultatai [65], [107] parodė, kad hibridinis algoritmas, derinantis evoliucinę globalią paiešką su efektyvia lokalia paieška, yra patikimiausias, deja, daugiausia skaičiavimo laiko reikalaujantis, algoritmas DS su Euklido atstumais. Genetinių algoritmų

DS su nestandartinėmis įtempimo funkcijomis privalumai analizuoti [50] darbe.

Bendrieji DS algoritmai gali būti taikomi miesto kvartalo atstumų atveju, jeigu jie nesiremia tikslo funkcijos diferencijuojamumu minimumo taške. Pavyzdžiui, standartinis Niutono metodas mažiausiųjų kvadratų įtempimo funkcijai su miesto kvartalo atstumais minimizuoti nėra tinkamas. Yra ir tik DS su miesto kvartalo atstumais išplėtotų metodų. DS su miesto kvartalo atstumais apžvalga pateikta [4] darbe. Čia nagrinėjami teoriniai klausimai, algoritmų raida, jų panaudojimas analizėje, palyginimai su kitais atstumais, sąryšiai su grafų teorijos modeliais.

Kombinatorinė paieška DS su miesto kvartalo atstumais pasiūlyta [75] darbe. Kombinatorinė lokali paieška yra naudojama gerai objektų eilės tvarkai kiekviename matmenyje nustatyti, o mažiausieji kvadratai – vaizdo taškų koordinatėms rasti remiantis objektų eilės tvarkomis.

Atstumų glaistymo (*distance smoothing*) metodas DS su miesto kvartalo atstumais pasiūlytas [62] darbe. Šis metodas leidžia išvengti lokalių minimumų. Jis apibendrintas [63] darbe Minkovskio atstumams ir pasiūlytas mažorizavimo algoritmas su monotoniškai nedidėjančių *Stress* įtempimo funkcijos reikšmių seka.

Euristinis atkaitinimo modeliavimo (*simulated annealing*) algoritmas dvimatėms skalėms su miesto kvartalo atstumais pasiūlytas [16] darbe. Euristika prasideda kiekvienos koordinačių ašies padalijimu į diskrečius vienodai nutolusius taškus. Atkaitinimo modeliavimo algoritmas ieško šiais taškais apibrėžtame tinklelyje mažiausiųjų kvadratų (3.3) arba mažiausiojo absoliučiojo nuokrypio (3.6) įtempimo funkcijos minimumo. Objektų eilės tvarkos kiekvienai koordinatei rastame sprendinyje naudojamos optimalioms koordinačių reikšmėms rasti kvadratiniu programavimu.

Dviejų lygmenų metodas dvimatėms skalėms su miesto kvartalo atstumais pasiūlytas [157] darbe, o vėliau apibendrintas DS esant bet kokiam d [158] darbe. Metodas remiasi dalimis kvadratine mažiausiųjų kvadratų įtempimo funkcijos su miesto kvartalo atstumais struktūra. Globalaus minimizavimo uždavinys keičiamas dviejų lygmenų minimizavimo uždaviniu. Viršutinio lygmens kombinatorinis optimizavimo uždavinys yra apibrėžtas d natūraliųjų skaičių $1,\ldots,m$ kėlinių rinkinių aibėje. Apatiniame lygmenyje sprendžiamas kvadratinio programavimo uždavinys su teigiamai apibrėžta kvadratine tikslo funkcija ir tiesiniais apribojimais, nustatančiais vaizdo taškų koordinačių reikšmių tvarkas, apibrėžtas kėlinių rinkiniais. Apatinio lygmens uždavinys

sprendžiamas naudojantis standartiniu kvadratinio programavimo algoritmu. Viršutinio lygmens uždavinys gali būti sprendžiamas garantuotais metodais, kai objektų skaičius m yra nedidelis, ir evoliucine paieška didesnėms duomenų aibėms. Lygiagretus dviejų lygmenų algoritmas, derinantis evoliucinę paiešką ir kvadratinį programavimą, yra pasiūlytas [156] darbe.

3.3.1. Evoliucinis algoritmas

Evoliucinės paieškos [113] idėja – saugoti geriausius tikslo funkcijos reikšmės prasme sprendinius, kuriuos sukryžminus galima sukurti geresnius sprendinius. Vienas iš esminių evoliucinės paieškos elementų – sprendinių kodavimas *chromosomomis*. Bendruosiuose tolydaus minimizavimo genetiniuose algoritmuose sprendiniai gali būti koduojami minimizavimo kintamųjų reikšmių vektoriumi. Pradinės chromosomos yra sugeneruojamos atsitiktinai.

Vėliau gali būti naudojamos kelios genetinės operacijos. *Mutavimo* metu atsitiktinai su tam tikra tikimybe pakeičiamas atsitiktinis genas. Nagrinėjamuoju atveju – tai kintamojo reikšmė. *Kryžminimo* metu iš dviejų populiacijos individų chromosomų sudaroma palikuonio chromosoma, paprastai viena dalis genų paimama iš vienos, o kita dalis – iš kitos tėvinių chromosomų. Prisitaikymas prie aplinkos gali būti modeliuojamas lokaliuoju minimizavimu. Kartais algoritmai, derinantys evoliucinę ir lokaliąją paieškas, vadinami *memetiniais*. Palikuonio gerumas yra apibrėžiamas tikslo funkcijos reikšme. Jei palikuonis yra geresnis už blogiausią populiacijos individą, jis pastarąjį pakeičia. Minimizavimas tęsiamas generuojant naujus palikuonis ir stabdomas po iš anksto nustatyto iteracijų skaičiaus arba laiko.

Daugiamačių skalių atveju tikslinga genais laikyti objektus vaizduojančius taškus. Tokiu atveju kryžminimo metu dalis taškų paimama iš vieno, o kita dalis – iš kito populiacijos individo, nesukryžminant skirtingų taškų. Tokiu atveju dviejų atsitiktinių esamos populiacijos individų su chromosomomis $\widehat{Y}$ ir $\breve{Y}$ palikuonio chromosoma randama remiantis formule

$$Y_i = \begin{cases} \widehat{Y}_i, & \text{jei } i \le \beta, i \ge \gamma, \\ \breve{Y}_i, & \text{jei } \beta < i < \gamma, \end{cases} \tag{3.18}$$

čia $i = 1,\ldots,m$, β ir γ – du atsitiktiniai skaičiai iš aibės $\{1,\ldots,m\}$.

Evoliucinis algoritmas gali būti realizuojamas naudojantis toliau nurodytu algoritmu. Pradinių atsitiktinių sprendinių skaičius n_{init}, populiacijos didumas n_p, sustojimo sąlyga pagal laiką t_c yra algoritmo parametrai.

Evoliucinis algoritmas daugiamatėms skalėms

Pradiniai duomenys: n_{init}, n_p, t_c, m, d, $\delta_{ij}, w_{ij}, i, j = 1,\ldots,m$

Rezultatas: S^*, Y^*

1. Atsitiktinai sugeneruoti n_{init} tolygiai pasiskirsčiusių md ilgio vektorių ir atsiminti n_p geriausių įtempimo funkcijos reikšmės prasme ir taip suformuoti populiaciją.
2. Pagerinti populiaciją atliekant lokalią paiešką iš populiacijos individų vektorių, atnaujinti vektorių elementų reikšmes gautais minimumų taškais.
3. ***Kol*** nepasiektas laikas t_c:
   a) Atsitiktinai parinkti du esamos populiacijos individus, pavadinkime jų chromosomas $\hat{Y}$ ir $\check{Y}$.
   b) Sugeneruoti palikuonį atliekant (3.18) kryžminimą ir lokalią paiešką iš gauto vektoriaus Y, atnaujinti vektoriaus elementų reikšmes gautu minimumo tašku.
   c) ***Jei*** palikuonis geresnis įtempimo funkcijos reikšmės prasme už blogiausią populiacijos individą,
   d) ***tai*** jis pastarąjį pakeičia, t. y. blogiausią populiacijos individą simbolizuojančio vektorius elementų reikšmės atnaujinamos palikuonio vektoriaus elementų reikšmėmis.

3.3.2. Dviejų lygmenų minimizavimas

Mažiausiųjų kvadratų įtempimo funkcija DS su miesto kvartalo atstumais gali būti nediferencijuojama net minimumo taške (žr. 3.2 skyrelį), todėl minimizuoti tokią funkciją yra sudėtinga.

Standartinių lokaliųjų paieškų algoritmų, besiremiančių tikslo funkcijos diferencijuojamumu minimumo taške, taikymas globaliojo optimizavimo pagreitinimui yra netinkamas. Dėl to [157] darbe pasiūlytas dviejų lygmenų algoritmas DS su miesto kvartalo atstumais, kai $d = 2$. Vėliau [158] darbe šis algoritmas apibendrintas bet kokiam d. Algoritme naudojamasi kombinatorine globaliąja paieška ir lokaliąja paieška, sukurta būtent įtempimo funkcijai su miesto kvartalo atstumais remiantis dalimis kvadratine tokios funkcijos struktūra.

Kai yra naudojami miesto kvartalo atstumai $d_1(Y_i, Y_j)$ (žr. (3.1) formulę), (3.2) mažiausiųjų kvadratų įtempimo funkciją galima užrašyti

$$S(Y)=\sum_{i=1}^{m}\sum_{j=1}^{m}w_{ij}\left(\sum_{k=1}^{d}\left|y_{ik}-y_{jk}\right|-\delta_{ij}\right)^2. \tag{3.19}$$

Pažymėkime $A(P)$ briaunainį – aibę (md)-matėje erdvėje:

$$A(P)=\left\{Y\,\middle|\, y_{ik}\le y_{jk},\ \text{jei}\ p_{ki}<p_{kj}, i,j=1,\ldots,m, k=1,\ldots,d\right\}, \tag{3.20}$$

čia $P=(P_1,\ldots,P_d)$ yra rinkinys iš d kėlinių, $P_k=(p_{k1},p_{k2},\ldots,p_{km})$ – natūraliųjų skaičių $1,\ldots,m$ kėliniai.

Kai $Y\in A(P)$, (3.19) formulę galima perrašyti taip:

$$S(Y)=\sum_{i=1}^{m}\sum_{j=1}^{m}w_{ij}\left(\sum_{k=1}^{d}(y_{ik}-y_{jk})z_{kij}-\delta_{ij}\right)^2,$$

čia

$$z_{kij}=\begin{cases}1, & \text{jei}\ p_{ki}\ge p_{kj},\\ -1, & \text{jei}\ p_{ki}<p_{kj}.\end{cases}$$

Taigi, užuot skaičiavę absoliutųjį didumą, dauginame iš kintamojo. Briaunainyje, P yra pastovus, taigi pastovus ir z_{kij}. Dėl to funkcija $S(Y)$ briaunainyje $A(P)$ yra kvadratinė, o minimizavimo uždavinys

$$\min_{Y\in A(P)} S(Y)$$

yra kvadratinio programavimo uždavinys. Toliau detalizuosime jį.

Mažiausiųjų kvadratų įtempimo funkcija DS su miesto kvartalo atstumais briaunainyje $A(P)$ yra

$$\begin{aligned}S(Y)&=\sum_{i=1}^{m}\sum_{j=1}^{m}w_{ij}\left(\sum_{k=1}^{d}(y_{ik}-y_{jk})z_{kij}-\delta_{ij}\right)^2=\\&=\sum_{i=1}^{m}\sum_{j=1}^{m}w_{ij}\delta_{ij}^2-2\sum_{i=1}^{m}\sum_{j=1}^{m}w_{ij}\delta_{ij}\sum_{k=1}^{d}(y_{ik}-y_{jk})z_{kij}+\\&+\sum_{i=1}^{m}\sum_{j=1}^{m}w_{ij}\left(\sum_{k=1}^{d}(y_{ik}-y_{jk})z_{kij}\right)^2.\end{aligned}$$

Pirmasis paskutinės lygybės narys yra konstanta kintamųjų $y_{ik}, i = 1,\ldots,m, k = 1,\ldots,d,$ atžvilgiu ir įtakos minimizavimui neturi. Antrasis narys yra tiesinis. Jį galima perrašyti taip:

$$-2\sum_{i=1}^{m}\sum_{j=1}^{m} w_{ij}\delta_{ij}\sum_{k=1}^{d}(y_{ik}-y_{jk})z_{kij} =$$

$$= -2\sum_{i=1}^{m}\sum_{j=1}^{m} w_{ij}\delta_{ij}\sum_{k=1}^{d} y_{ik}z_{kij} + 2\sum_{i=1}^{m}\sum_{j=1}^{m} w_{ij}\delta_{ij}\sum_{k=1}^{d} y_{jk}z_{kij} =$$

$$= -4\sum_{i=1}^{m}\sum_{j=1}^{m} w_{ij}\delta_{ij}\sum_{k=1}^{d} y_{ik}z_{kij} =$$

$$= -4\sum_{k=1}^{d}\sum_{i=1}^{m} y_{ik}\sum_{j=1}^{m} w_{ij}\delta_{ij}z_{kij}.$$

Trečiasis narys yra kvadratinis ir gali būti perrašytas:

$$\sum_{i=1}^{m}\sum_{j=1}^{m} w_{ij}\left(\sum_{k=1}^{d}(y_{ik}-y_{jk})z_{kij}\right)^2 =$$

$$= \sum_{i=1}^{m}\sum_{j=1}^{m} w_{ij}\sum_{k=1}^{d}\sum_{l=1}^{d}(y_{ik}-y_{jk})(y_{il}-y_{jl})z_{kij}z_{lij} =$$

$$= \sum_{k=1}^{d}\sum_{l=1}^{d}\sum_{i=1}^{m}\sum_{j=1}^{m}(y_{ik}y_{il}-y_{ik}y_{jl}-y_{jk}y_{il}+y_{jk}y_{jl})w_{ij}z_{kij}z_{lij} =$$

$$= \sum_{k=1}^{d}\sum_{l=1}^{d}\sum_{i=1}^{m}\sum_{j=1}^{m} y_{ik}y_{il}w_{ij}z_{kij}z_{lij} + \sum_{k=1}^{d}\sum_{l=1}^{d}\sum_{i=1}^{m}\sum_{j=1}^{m} y_{jk}y_{jl}w_{ij}z_{kij}z_{lij} -$$

$$-\sum_{k=1}^{d}\sum_{l=1}^{d}\sum_{i=1}^{m}\sum_{j=1}^{m} y_{ik}y_{jl}w_{ij}z_{kij}z_{lij} - \sum_{k=1}^{d}\sum_{l=1}^{d}\sum_{i=1}^{m}\sum_{j=1}^{m} y_{jk}y_{il}w_{ij}z_{kij}z_{lij} =$$

$$= 2\sum_{k=1}^{d}\sum_{l=1}^{d}\sum_{i=1}^{m}\sum_{j=1}^{m} y_{ik}y_{il}w_{ij}z_{kij}z_{lij} - 2\sum_{k=1}^{d}\sum_{l=1}^{d}\sum_{i=1}^{m}\sum_{j=1}^{m} y_{ik}y_{jl}w_{ij}z_{kij}z_{lij} =$$

$$= 2\sum_{k=1}^{d}\sum_{l=1}^{d}\sum_{i=1}^{m} y_{ik}y_{il}\sum_{j=1}^{m} w_{ij}z_{kij}z_{lij} -$$

$$-2\sum_{k=1}^{d}\sum_{l=1}^{d}\sum_{i=1}^{m}\sum_{j=1,\,j\neq i}^{m} y_{ik}y_{jl}w_{ij}z_{kij}z_{lij} - 2\sum_{k=1}^{d}\sum_{l=1}^{d}\sum_{i=1}^{m} y_{ik}y_{il}w_{ii}z_{kii}z_{lii} =$$

$$= 2\sum_{k=1}^{d}\sum_{l=1}^{d}\sum_{i=1}^{m} y_{ik}y_{il}\sum_{t=1,t\neq i}^{m} w_{it}z_{kit}z_{lit} - 2\sum_{k=1}^{d}\sum_{l=1}^{d}\sum_{i=1}^{m}\sum_{j=1,\,j\neq i}^{m} y_{ik}y_{jl}w_{ij}z_{kij}z_{lij}.$$

Tiesinį ir kvadratinį narius galima padalyti iš 4, nes tai neturės įtakos minimizavimui.

Kadangi $Y \in A(P)$, tai (3.20) formulėje nurodytas sąlygas

$$y_{ik} \le y_{jk}, \text{ jei } p_{ki} < p_{kj},\ i,j = 1,\ldots,m,\ k = 1,\ldots,d,$$

galima pakeisti joms ekvivalenčiu apribojimu

$$y_{\{j|p_{kj}=i\}k} \le y_{\{j|p_{kj}=i+1\}k},\ k = 1,\ldots,d,\ i = 1,\ldots,m-1.$$

Todėl

$$y_{\{j|p_{kj}=i+1\}k} - y_{\{j|p_{kj}=i\}k} \ge 0,\ k = 1,\ldots,d,\ i = 1,\ldots,m-1.$$

Įtempimo funkcija ir jos apribojimai yra invariantiniai perkėlimo atžvilgiu (nesikeičia pridedant vienodas reikšmes prie visų $y_{ik}, i = 1,\ldots,m$). Tai yra nepalanku minimizavimui, bet to galima išvengti centruojant sprendinį pagal kiekvieną koordinatę (žr. (3.7) formulę).

Taigi kvadratinio programavimo uždavinys yra

$$\min\left[-\sum_{k=1}^{d}\sum_{i=1}^{m} y_{ik}\sum_{j=1}^{m} w_{ij}\delta_{ij}z_{kij} + \right.$$

$$\left. +\frac{1}{2}\left(\sum_{k=1}^{d}\sum_{l=1}^{d}\sum_{i=1}^{m} y_{ik}y_{il}\sum_{t=1,t\neq i}^{m} w_{it}z_{kit}z_{lit} - \sum_{k=1}^{d}\sum_{l=1}^{d}\sum_{i=1}^{m}\sum_{j=1,\,j\neq i}^{m} y_{ik}y_{jl}w_{ij}z_{kij}z_{lij}\right)\right],$$

esant apribojimams:

$$\sum_{i=1}^{m} y_{ik} = 0,\ k = 1,\ldots,d,$$

$$y_{\{j|p_{kj}=i+1\}k} - y_{\{j|p_{kj}=i\}k} \geq 0,\ k=1,\ldots,d,\ i=1,\ldots,m-1.$$

Šis uždavinys gali būti užrašytas matricine forma:

$$\min\left(-B^T Y + \frac{1}{2} Y^T D Y\right), \tag{3.21}$$

esant apribojimams:

$$A^0 Y = 0, \tag{3.22}$$

$$A^k Y \geq 0,\ k=1,\ldots,d. \tag{3.23}$$

Čia Y yra md ilgio vektorius, apimantis visas vaizdo taškų koordinačių reikšmes, iš viso md reikšmių; B yra md ilgio vektorius, D yra kvadratinė $(md)\times(md)$ dydžio matrica (po eilutę ir stulpelį kiekvienai vaizdo taškų koordinačių reikšmei); A^0 yra $d\times(md)$ dydžio (d eilučių ir md stulpelių) matrica, o $A^k, k=1,\ldots,d,$ yra $(m-1)\times(md)$ matricos, kurių elementai yra:

$$B_{km-m+i} = \sum_{j=1}^{m} w_{ij}\delta_{ij} z_{kij},\quad k=1,\ldots,d,\ i=1,\ldots,m,$$

$$D_{(km-m+i)(lm-m+j)} = \begin{cases} \sum_{t=1,t\neq i}^{m} w_{it} z_{kit} z_{lit}, & \text{jei } i=j, \\ -w_{ij} z_{kij} z_{lij}, & \text{jei } i\neq j, \end{cases}$$

čia $k,l=1,\ldots,d,\ i,j=1,\ldots,m,$

$$A^0_{kj} = \begin{cases} 1, \text{ jei } j = km-m+1,\ldots,km, \\ 0 \text{ kitu atveju,} \end{cases}$$

čia $k=1,\ldots,d,\ j=1,\ldots,md,$

$$A^k_{ij} = \begin{cases} 1, & \text{jei } p_{k(j-km+m)} = i+1, \\ -1, & \text{jei } p_{k(j-km+m)} = i, \\ 0 & \text{kitu atveju,} \end{cases} \tag{3.24}$$

čia $k=1,\ldots,d,\ i=1,\ldots,m-1,\ j=1,\ldots,md.$

Briaunainis $A(P)$ apibrėžtas tiesiniais nelygybiniais (3.23) apribojimais. Tiesiniai lygybiniai (3.22) apribojimai užtikrina sprendinio centravimą pagal kiekvieną koordinatę, kad būtų išvengta invariantiškumo perkėlimo atžvilgiu.

Uždavinys (3.21)–(3.24) gali būti sprendžiamas standartiniu kvadratinio programavimo metodu. Tačiau kvadratinio programavimo uždavinio sprendinys nebūtinai yra įtempimo funkcijos minimizavimo uždavinio lokalaus minimumo taškas. Jei kvadratinio programavimo uždavinio sprendinys Y yra briaunainio $A(P)$ paviršiuje, tikėtina, kad lokalaus minimumo taškas yra gretimame briaunainyje. Dėl to paieška gali būti tęsiama sprendžiant kvadratinio programavimo uždavinį gretimame briaunainyje. Ieškant tokio briaunainio, kėliniai P turi būti atnaujinti atsižvelgiant į aktyvius nelygybinius apribojimus, t. y. tokius, kurie tenkinami kaip lygybiniai. Jei $i,\ldots,j, i<j$, nelygybiniai apribojimai $A^k Y \geq 0$ yra aktyvūs, $i \leq p_{kt} \leq j+1$ turi būti pakeisti į $i+j+1-p_{kt}$. Minimizavimas tęsiamas, kol randamos geresnės reikšmės ir bent keli nelygybiniai apribojimai yra aktyvūs.

Minimizavimo uždavinys sprendžiamas naudojantis dviejų lygmenų minimizavimo algoritmu. Viršutiniame lygmenyje sprendžiamas kombinatorinis uždavinys, o apatiniame – kvadratinio programavimo uždavinys:

$$\min_{P} S(P), \tag{3.25}$$

esant apribojimui

$$S(P) = \min_{Y \in A(P)} S(Y). \tag{3.26}$$

Apatinio lygmens (3.26) uždavinį galima spręsti naudojantis standartiniu kvadratinio programavimo algoritmu. Viršutinio lygmens (3.25) uždavinio tikslo funkcija yra apibrėžta d natūraliųjų skaičių $1,\ldots,m$ kėlinių rinkinių aibėje. Šio kombinatorinio uždavinio leistinų sprendinių skaičius yra $(m!)^d$. Šis uždavinys gali būti sprendžiamas įvairiai, pavyzdžiui naudojantis:

- sprendinių perrinkimu;
- šakų ir rėžių algoritmu;
- atsitiktine paieška;
- lokaliomis paieškomis iš atsitiktinių sprendinių;
- evoliucine paieška;
- kitais algoritmais.

Taikant sprendinių perrinkimo algoritmą, DS uždavinį galima išspręsti garantuotai, tačiau tik mažesnės apimties. Didesnius uždavinius pavyksta įveikti naudojantis šakų ir rėžių algoritmu. Atsitiktinės paieškos atveju, kėliniai P yra generuojami atsitiktinai ir tada sprendžiamas apatinį lygmenį atitinkantis

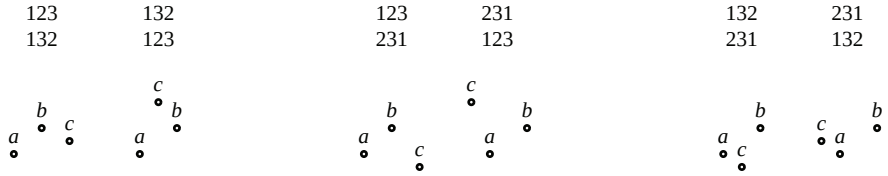
kvadratinio programavimo uždavinys. Lokalios paieškos gali būti atliekamos per gretimus briaunainius. Gali būti naudojama evoliucinė arba kitokia metaeuristinė paieška.

3.3.3. Sprendinių perrinkimas kombinatoriniame algoritme

Daugiamačių skalių su miesto kvartalo atstumais sprendiniai yra invariantiniai koordinačių ašių krypčių pakeitimo ir koordinačių sukeitimo atžvilgiu. Pavyzdžiui, kai $d = 2$, invariantiniai koordinačių ašių krypčių pakeitimo atžvilgiu sprendiniai pavaizduoti 3.1 paveiksle. Kiekvienas skaitmuo atitinka p_{ki} reikšmę, $k = 1,\ldots,d, i = 1,\ldots,m.$ Mažesnio k kėliniai yra vaizduojami aukščiau. Pavyzdžiui, kėliniai „12/21“ reiškia, kad pirmojo objekto pirmosios koordinatės reikšmė nebus didesnė už antrojo – objektas *a* nebus dešiniau už *b*, ir pirmojo objekto antrosios koordinatės reikšmė nebus mažesnė už antrojo – objektas *a* nebus žemiau už *b*. Ekvivalenčių sprendinių koordinačių ašių krypčių pakeitimo atžvilgiu yra 2^d.

Invariantinių koordinačių ašių sukeitimo atžvilgiu sprendinių poros, kai $d = 2,$ pavaizduotos 3.2 paveiksle. Tokių ekvivalenčių sprendinių grupes sudaro $d!$ sprendinių. Sprendžiant DS uždavinius, užtenka rasti vieną iš ekvivalenčių sprendinių. Unikalių sprendinių skaičių galima rasti įvertinus invariantiškumą.

3.1 pav. Invariantiniai koordinačių ašių krypčių pakeitimo atžvilgiu sprendiniai ir juos aprašantys kėliniai, kurie nustato objektų *a* ir *b* koordinačių reikšmių tvarką



3.2 pav. Invariantinių koordinačių ašių sukeitimo atžvilgiu sprendinių poros ir juos aprašantys kėliniai, kurie nustato objektų *a*, *b* ir *c* koordinačių reikšmių tvarką

Įvertinus invariantiškumą koordinačių ašių krypčių pakeitimo atžvilgiu, unikalių sprendinių skaičius sumažinamas iki $(m!)^d / 2^d = (m!/2)^d$. Įvertinus invariantiškumą koordinačių sukeitimo atžvilgiu, unikalių sprendinių skaičius sumažinamas iki maždaug $(m!/2)^d / d!$. Pažymėkime $u = n!/2$. Kai $d = 1$, unikalių sprendinių skaičius yra u. Taigi šiuo atveju invariantiškumo koordinačių ašių sukeitimo atžvilgiu nėra, nes vienmatės skalės atveju yra tik viena koordinačių ašis. Kai $d = 2$, unikalių sprendinių skaičius yra $(u^2 + u)/2$. Šiuo atveju ekvivalenčių sprendinių poroms atstovauja vienas sprendinys. Tačiau dalis sprendinių yra be porų, pavyzdžiui, „123/123“, „132/132“ ir „231/231“, nes visų koordinačių kėliniai yra vienodi. Kai $d = 3$, unikalių sprendinių skaičius yra $(u^3 + 3u^2 + 2u)/6$, o kai $d = 4$, jų skaičius yra $(u^4 + 6u^3 + 11u^2 + 6u)/24$.

Perrenkant unikalius sprendinius, nėra svarbu, kokia tvarka jie perrenkami, bet svarbu užtikrinti, kad tik vienas iš ekvivalenčių viršutinio lygmens kombinatorinio uždavinio sprendinių būtų įvertintas sprendžiant apatinio lygmens kvadratinio programavimo uždavinį. Vienas iš galimų unikalių sprendinių perrinkimo algoritmų daugiamatėms skalėms pateikiamas šiame skyrelyje. Jis pasirinktas todėl, kad panašus į tolesniame skyrelyje parodytą šakų ir rėžių algoritmą.

Algoritme priskyrimo veiksmas žymimas rodykle „$\leftarrow$“. Paieška pradedama nuo kėlinių $p_{ki} = i, i = 1,\ldots,m, k = 1,\ldots,d$. Vėliau paeiliui keičiami vieną iš koordinačių atitinkantys kėliniai. Keičiant kėlinį, paskutinio objekto numeris mažinamas sukeičiant su didžiausią numerį iki jo turėjusiu objektu (tai daroma apatinėje algoritmo dalyje), vėliau su sekančiu ir t. t., kol paskutinio objekto numeris tampa lygus vienetui. Kai paskutinio objekto numerio jau nebegalima sumažinti, priešpaskutinio objekto numeris mažinamas sukeičiant jį su didžiausią numerį iki jo turėjusiu objektu, o paskutiniam objektui priskiriamas didesnis numeris. Pagrindinis algoritmo ciklas kartojamas, kol $j > 2$. Taip užtikrinama, kad tik vieno iš sprendinių, invariantinių koordinačių ašių krypties pakeitimo atžvilgiu, bus ieškoma, nes pirmąjį ir antrąjį objektus vaizduojančių taškų pozicijos nebus sukeistos (žr. 3.1 pav.).

Taip suformuojama tokia kėlinių tvarka: natūraliųjų skaičių 1, 2, 3 kėlinių – „123“ $\prec$ „132“ $\prec$ „231“; natūraliųjų skaičių 1,...,4 kėlinių – „1234“ $\prec$ „1243“ $\prec$ „1342“ $\prec$ „2341“ $\prec$ „1324“ $\prec$ „1423“ $\prec$ „1432“ $\prec$ „2431“ $\prec$ „2314“ $\prec$ „2413“ $\prec$ „3412“ $\prec$ „3421“; ir taip toliau.

Kad tik vieno iš sprendinių, invariantinių koordinačių sukeitimo atžvilgiu, būtų ieškoma, algoritmo vidurinėje dalyje atliekamas patikrinimas. Kėlinys P_k negali būti ankstesnis už P_l, kai $k > l$, dėl to $P_k \prec P_{k-1}$ negali būti. Todėl daliniai sprendiniai „132/123“, „231/123“ ir „231/132“ neleistini, nes jie yra ekvivalentūs atitinkamai „123/132“, „123/231“ ir „132/231“ (žr. 3.2 pav.).

Unikalių sprendinių perrinkimo algoritmas daugiamatėms skalėms

Pradiniai duomenys: $m,\ d,\ \delta_{ij}, w_{ij}, i, j = 1,\ldots,m$

Rezultatas: $S^*,\ Y^*$

$p_{ki} \leftarrow i,\ i = 1,\ldots,m,\ k = 1,\ldots,d\ ;\ j \leftarrow m+1\ ;\ k \leftarrow d+1\ ;\ S^* \leftarrow \infty$
Kol $j > 2$
    Jei $j > m$
        Jei $\min\limits_{Y \in A(P)} S(Y) < S^*$ // Sprendinio įvertinimas
            $S^* \leftarrow \min\limits_{Y \in A(P)} S(Y)\ ;\ Y^* \leftarrow \arg\min\limits_{Y \in A(P)} S(Y)$
        $j \leftarrow m\ ;\ k \leftarrow d$
    Jei $j > 2$ // Formuojamas kitas kėlinių rinkinys
        Jei $p_{kj} = 0$
            $p_{kj} \leftarrow j$
            Jei $k > 1$ ir $P_k \prec P_{k-1}$ // Ekvivalentumo patikrinimas
                $p_{ki} \leftarrow p_{(k-1)i}, i = 1,\ldots,j$
            $k \leftarrow k+1$
        Kitu atveju
            $p_{kj} \leftarrow p_{kj} - 1$
            Jei $p_{kj} = 0$
                $p_{ki} \leftarrow p_{ki} - 1, i = 1,\ldots,j-1\ ;\ k \leftarrow k-1$
                Jei $k < 1$: $j \leftarrow j-1\ ;\ k \leftarrow d$
            Kitu atveju
                $i \leftarrow j-1$ // Rasti *i* tokį, kad
                Kol $p_{ki} \neq p_{kj}$: $i \leftarrow i-1$ // $p_{ki} = p_{kj}$
                $p_{ki} \leftarrow p_{ki} + 1\ ;\ k \leftarrow k+1$

Šiuolaikišku asmeniniu kompiuteriu taikant šį algoritmą galima garantuotai įveikti DS uždavinius, kai $m = 8$ ir $d = 2$, kuriems būtina garantuotai išspręsti 16 kintamųjų globalaus minimizavimo uždavinį su ne visur diferencijuojama tikslo funkcija.

Kartais daugiamačiai duomenys gali būti simetriniai, pavyzdžiui, kai objektus sukeitus skirtingumo reikšmės nesikeičia. Tokiais atvejais egzistuoja sprendiniai, invariantiniai duomenų objektų sukeitimo atžvilgiu. Uždavinių sprendimo laiką galima gerokai sutrumpinti susiaurinus paiešką taip, kad būtų ieškoma tik vieno iš ekvivalenčių sprendinių [160]. Tolydaus minimizavimo atveju galima nustatyti apribojimus simetrinių objektų vaizdo taškų pirmosios koordinatės reikšmių tvarkai. Kombinatoriniame viršutinio lygmens algoritme galima leisti ne visus pirmosios koordinatės reikšmių kėlinius P_1.

Pavyzdžiui, visos standartinio simplekso (žr. 3.4 skyrelį) viršūnės yra simetrinės. Jeigu bet kuri viršūnė būtų sukeista su bet kuria kita, visų viršūnių skirtingumai būtų tokie pat: $\delta_{ij} = 1, i \neq j$. Taigi yra daug ekvivalenčių sprendinių, kuriuose keičiasi tik viršūnių numeracija. Toks simpleksas turi $m!$ skirtingų numeracijų. Paieškos sritį galima apriboti taip, kad tik vieno iš ekvivalenčių sprendinių būtų ieškoma, nustačius objektus vaizduojančių taškų pirmosios koordinatės reikšmių tvarką. Tolydaus minimizavimo atveju galima pareikalauti, kad galiotų nelygybės $y_{11} \le y_{21} \le \cdots \le y_{m1}$, kurios kombinatorinio minimizavimo algoritme atitiktų vienintelį leistiną pirmosios koordinatės kėlinį $P_1 = (1,2,\ldots,m)$. Tokiu atveju unikalių sprendinių skaičius lygus 1, kai $d = 1$; u, kai $d = 2$; $(u^2 + u)/2$, kai $d = 3$; ir $(u^3 + 3u^2 + 2u)/6$, kai $d = 4$.

Išvengus invariantiškumo objektų sukeitimo atžvilgiu, reikalingų išspręsti apatinio lygmens kvadratinio programavimo uždavinių skaičius sumažinamas maždaug u/d kartų. Kvadratinio programavimo uždavinių skaičius yra toks pat kaip $d - 1$ atveju, kai invariantiškumo neišvengiama. Nors kvadratinio programavimo uždaviniai yra didesnės apimties dėl didesnio d, per panašų laiką galima išspręsti d-matės skalės uždavinius vietoj $(d-1)$-matės. Kai $d = 2$, šiuolaikišku asmeniniu kompiuteriu per priimtiną laiką galima išspręsti uždavinį, kuriame $m = 12$ vietoj $m = 8$, o kai $d = 3$, – $m = 8$ vietoj $m = 6$. Taigi 24 kintamųjų globalaus minimizavimo uždaviniai su ne visur diferencijuojama tikslo funkcija yra garantuotai išsprendžiami.

Vienetinio simplekso (žr. 3.4 skyrelį) viršūnės ne koordinačių pradžioje taip pat yra simetrinės. Jeigu tokia viršūnė būtų sukeista su kita viršūne ne

koordinačių pradžioje, viršūnes atitinkančios skirtingumų matricos eilutės ir stulpeliai būtų sukeisti, bet skirtingumų matrica liktų ta pati. Taigi viršūnių skirtingumai liktų tokie pat. Vadinasi, ir šiuo atveju yra ekvivalenčių sprendinių, tik šis simpleksas turi $(m-1)!$ skirtingų numeracijų, atitinkančių tą pačią skirtingumų matricą. Vieno iš ekvivalenčių sprendinių būtų galima ieškoti, jei paieškos sritis būtų apribota, nurodžius viršūnes ne koordinačių pradžioje vaizduojančių taškų pirmosios koordinatės reikšmių seką.

Tolydaus minimizavimo atveju gali būti įvesti apribojimai $y_{21} \le y_{31} \le \cdots \le y_{m1}$, kurie kombinatorinio minimizavimo algoritme atitiktų $\lceil m/2 \rceil$ leistinų pirmosios koordinatės kėlinių $P_1 = (l, l+1, \ldots, m, 1, 2, \ldots, l-1)$, $l \le m/2$. Čia $\lceil m/2 \rceil$ žymi mažiausią sveikąjį skaičių, ne mažesnį už $m/2$ – ieškoma vieno iš sprendinių, invariantinių pirmosios koordinačių ašies krypties pakeitimo atžvilgiu. Kai $d = 1$, unikalių sprendinių skaičius yra lygus $\lceil m/2 \rceil$. Kai $d > 1$, unikalių sprendinių skaičius, palyginti su standartinio simplekso atveju, apytikriai yra didesnis $m/3$ kartų. Taigi išvengus invariantiškumo objektų sukeitimo atžvilgiu, *d*-matės skalės uždavinius išspręsti trunka nedaug ilgiau kaip $(d-1)$-matės, kai invariantiškumo neišvengiama. Kaip ir standartinio simplekso atveju, kai $d = 2$, per priimtiną laiką galima išspręsti uždavinį, kuriame $m = 12$, o kai $d = 3$, pavyksta išspręsti uždavinį, kuriame $m = 8$.

Daugiamačio kubo (žr. 3.4 skyrelį) viršūnės taip pat yra simetrinės. Tą pačią skirtingumų matricą atitinkančių viršūnių numeracijų skaičius yra $m \times n!$. Galima laisvai pasirinkti vieną iš viršūnių, tada laisvai pasirinkti vieną iš koordinačių ašių daugiamatėje duomenų erdvėje, tada vieną iš likusių koordinačių ašių ir taip toliau. Sunkiau aprašyti, kaip reikėtų sukeisti objektus, kad skirtingumų matrica nepasikeistų, nes tokiame procese dalyvautų daug eilučių ir stulpelių vienu metu.

Apribojus paieškos sritį, daug ekvivalenčių sprendinių būtų galima neieškoti. Galima pareikalauti, kad vieną iš viršūnių, pavyzdžiui, esančią koordinačių pradžioje, vaizduojantis taškas būtų kairiausias. Tolydaus optimizavimo atveju galima įvesti apribojimus $y_{11} \le y_{i1}, i = 2, \ldots, m$, kurie kombinatorinio minimizavimo algoritme atitiktų kėlinius su fiksuotu $p_{11} = 1$. Toks paprastas apribojimas gerokai sumažina apatinio lygmens kvadratinio programavimo uždavinių skaičių.

Kadangi DS yra sudėtingi globalaus minimizavimo uždaviniai, jų sprendimas reikalauja nemažai laiko. Naudojantis lygiagrečiaisiais

skaičiavimais, galima sutrumpinti sprendimo laiką ir per priimtiną laiką išspręsti didesnės apimties uždavinius.

Lygiagrečiajame viršutinio lygmens kombinatorinio uždavinio unikalių sprendinių perrinkimo algoritme kiekvienas procesorius vykdo tokį pat sprendinių generavimo algoritmą, tačiau tik kas p-ajam sprendiniui yra sprendžiamas apatinio lygmens kvadratinio programavimo uždavinys. Čia p yra procesorių skaičius. Pirmasis procesorius sprendžia apatinio lygmens kvadratinio programavimo uždavinį pirmajam, $(p+1)$-ajam ir t. t.; antrasis procesorius – antrajam, $(p+2)$-ajam ir t. t.; o p-asis procesorius – p-ajam, $2p$-ajam ir t. t. sugeneruotam unikaliam viršutinio lygmens kombinatorinio uždavinio sprendiniui. Daroma prielaida, kad unikalių viršutinio lygmens kombinatorinio uždavinio sprendinių generavimas užima kur kas mažiau laiko negu apatinio lygmens kvadratinio programavimo uždavinių sprendimas. Skirtingų procesorių rezultatai surenkami pabaigus skaičiavimus. Tokio algoritmo spartinimas yra beveik lygus procesorių skaičiui, o lygiagretinimo efektyvumas – artimas vienetui. Šiuo algoritmu 10 kompiuterių klasteryje galima per kelias paras garantuotai išspręsti DS uždavinius, kai $d=2$ ir $m=9$, nepriklausomai nuo duomenų aibės. Globalaus optimizavimo uždavinys su 18 kintamųjų yra pakankamai didelis, norint garantuotai jį išspręsti.

3.3.4. Šakų ir rėžių algoritmas

Pagrindinė šakų ir rėžių algoritmo idėja – išvengti dalies leistinųjų sprendinių išreikštinio perrinkimo nustatant leistinosios srities posričius, kuriuose globalaus minimumo negali būti. Tikslo funkcijos apatinis rėžis posrityje turi būti įvertintas ir palygintas su tuo metu žinoma geriausia tikslo funkcijos reikšme. Jeigu rėžis yra blogesnis už žinomą reikšmę, posrityje geresnės reikšmės nebus ir dėl to tolesnė paieška šiame posrityje nevykdoma.

Paieškos procesas gali būti vaizduojamas paieškos medžiu, kurio šaknis žymi leistinąją sritį, o viršūnės (išsišakojimai) – jos posričius. Paieška vykdoma šakojant paieškos medį – dalijant pasirinktą posritį į kelis. Jeigu įvertinus rėžį yra nustatoma, kad posrityje globalaus minimumo negali būti, posritį žyminti viršūnė nešakojama – šaka nukertama. Kuo anksčiau šakos yra nukertamos, tuo mažesnis leistinų sprendinių skaičius yra patikrinamas išreikštiniu būdu ir minimizavimui reikia mažiau skaičiavimų.

Šakų ir rėžių algoritmus sudaro inicializavimo, išrinkimo, dalijimo ir rėžių skaičiavimo taisyklės. Inicializavimo etape leistinoji sritis yra padengiama nurodytos formos posričiais. Toliau vykdomas ciklas: iš kandidatų aibės

(nenukirstų dar nepatikrintų viršūnių) išrenkamas ir padalijamas posritis; apskaičiuojami tikslo funkcijos minimumo rėžiai naujai gautuose posričiuose; posričiai, kuriuose minimumo taškas negali būti, pašalinami, o nepašalinti posričiai įtraukiami į kandidatų aibę.

Padengimo, išrinkimo, padalijimo ir rėžių skaičiavimo taisyklės priklauso nuo konkretaus algoritmo. Padengimo ir padalijimo taisyklės priklauso nuo naudojamų posričių formos. Naudojama viena iš trijų išrinkimo taisyklių: geriausiojo išrinkimo – išrenkamas kandidatas su geriausiu įverčiu (su mažiausiu apatiniu tikslo funkcijos rėžiu); gilyn – išrenkamas jauniausias kandidatas (toliausias nuo paieškos medžio šaknies); platyn – išrenkamas vyriausias kandidatas (arčiausias prie paieškos medžio šaknies).

Rėžių įvertinimas yra esminė šakų ir rėžių algoritmų dalis. Jei rėžiai yra nepakankamai siauri, gali tekti perrinkti visus leistinus sprendinius išreikštinai, taigi algoritmas susivestų į leistinų sprendinių perrinkimą. Tai yra nepageidaujama, nes dažniausiai leistinų sprendinių skaičius auga eksponentiškai priklausomai nuo uždavinio dydžio. Rėžiai sudaromi atsižvelgiant į tikslo funkciją ir naudojamų posričių tipą.

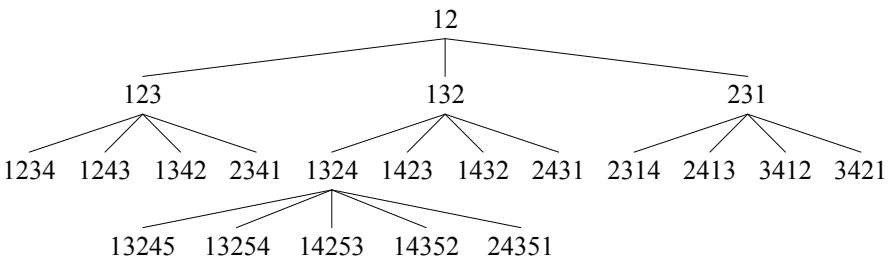
Sudarant šakų ir rėžių algoritmą DS, galima pastebėti, kad įtraukus papildomą objektą įtempimo funkcijos reikšmė nesumažėja – ji gali tik padidėti arba idealiu atveju likti nepakitusi. Dėl to posritis gali būti aprašomas mažesniu uždaviniu, kuriame yra tik dalis objektų, o įtempimo funkcijos apatinis rėžis tokiame posrityje gali būti įvertintas radus mažesnio uždavinio įtempimo funkcijos minimumą.

Tai įvertinus, galima sudaryti šakų ir rėžių algoritmą viršutinio lygmens kombinatoriniam uždaviniui spręsti DS su miesto kvartalo atstumais. Posritis yra aprašomas daliniu sprendiniu, apibrėžtu d natūraliųjų skaičių $1,\ldots,\bar{m}$ kėliniais, čia $\bar{m} < m$ yra vertinamų objektų skaičius. Įtempimo funkcijos apatinis rėžis tokiame posrityje yra dalinės įtempimo funkcijos (kuria įvertinama tik $\bar{m}$ objektų) reikšmė atitinkamo kvadratinio programavimo uždavinio minimumo taške:

$$\min_{\bar{Y}\in A(\bar{P})} \bar{S}(\bar{Y}) = \min_{\bar{Y}\in A(\bar{P})} \sum_{i=1}^{\bar{m}} \sum_{j=1}^{\bar{m}} w_{ij} \left(\sum_{k=1}^{d} (y_{ik} - y_{jk}) z_{kij} - \delta_{ij} \right)^2, \qquad (3.27)$$

čia $\bar{Y} = (Y_1, Y_2, \ldots, Y_{\bar{m}})$, $\bar{P} = (\bar{P}_1, \ldots, \bar{P}_d)$, $\bar{P}_k = (p_{k1}, p_{k2}, \ldots, p_{k\bar{m}})$. Naujai įtraukiami paieškoje objektai turi nepakeisti anksčiau nustatytos $\bar{m}$ objektų koordinačių reikšmių tvarkos, tada ir įtempimo funkcijos reikšmė nesumažės.

Šakų ir rėžių algoritmo DS paieškos medis, kai $d = 1,$ pavaizduotas 3.3 paveiksle. Kiekvienas skaitmuo atitinka p_{1i} reikšmę, $i = 1,\ldots,\bar{m}.$ Kad tik vieno iš invariantinių koordinačių ašies krypties pakeitimo atžvilgiu sprendinių būtų ieškoma, medžio šaknis yra „12“. Taigi pirmąjį objektą vaizduojantis taškas niekada nebus dešiniau antrąjį objektą vaizduojančio taško. Šio dalinio sprendinio įtempimo funkcijos rėžis nėra vertinamas, nes jis aprašo unikalių sprendinių aibę. Trečiąjį objektą vaizduojantis taškas gali būti įterpiamas dešiniau pirmųjų dviejų, tarp jų arba kairiau jų. Tokie įterpimai aprašomi trimis paieškos medžio šakomis, kurių viršūnės yra „123“, „132“ ir „231“.



3.3 pav. Šakų ir rėžių algoritmo DS paieškos medis, kai $d = 1$

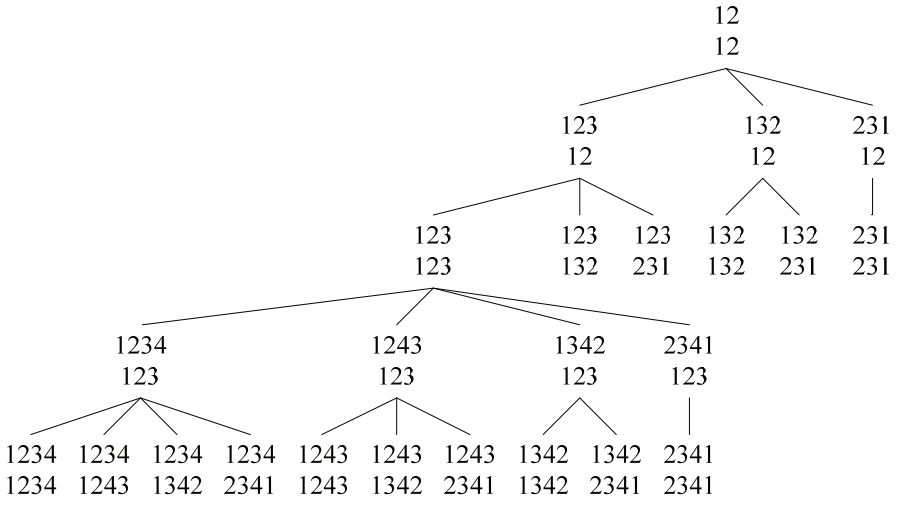
Nors pirmąjį ir antrąjį objektus vaizduojančių taškų koordinačių reikšmių tvarkos numeriai kėliniuose, įterpus naują objektą, gali pasikeisti (į „13“ arba „23“), jų tvarka nesikeis ($1 < 2,\ 1 < 3$ ir $2 < 3$). Ketvirtąjį objektą vaizduojantį tašką bus galima įterpti keturiuose intervaluose, dėl to paieškos medyje yra keturios įterpimą vaizduojančios šakos. Nors anksčiau priskirtų objektų tvarkos numeriai gali pasikeisti, bet jų tvarka nesikeis. Penktojo objekto įterpimas vaizduojamas penkiomis šakomis ir t. t.

Jeigu dalinės įtempimo funkcijos reikšmė dalinį sprendinį atitinkančio apatinio lygmens kvadratinio uždavinio minimumo taške yra didesnė už anksčiau įvertintą įtempimo funkcijos reikšmę pilnąjį sprendinį atitinkančio apatinio lygmens kvadratinio uždavinio minimumo taške, daliniu sprendiniu aprašomoje sprendinių aibėje geresnio sprendinio nebus. Tokiu atveju dalinį sprendinį vaizduojančią šaką galima nukirsti ir vėliau paieškoje nebešakoti. Taigi kai $d = 1,$ unikalių sprendinių skaičius yra $m!/2.$

Paieškos medis, kai $d = 2,$ yra pavaizduotas 3.4 paveiksle. Kiekvienas skaitmuo atitinka p_{ki} reikšmę, $k = 1,\ldots,d, i = 1,\ldots,\bar{m}.$ Mažesnio k kėliniai yra vaizduojami aukščiau. Kad tik vieno iš invariantinių koordinačių ašių krypčių

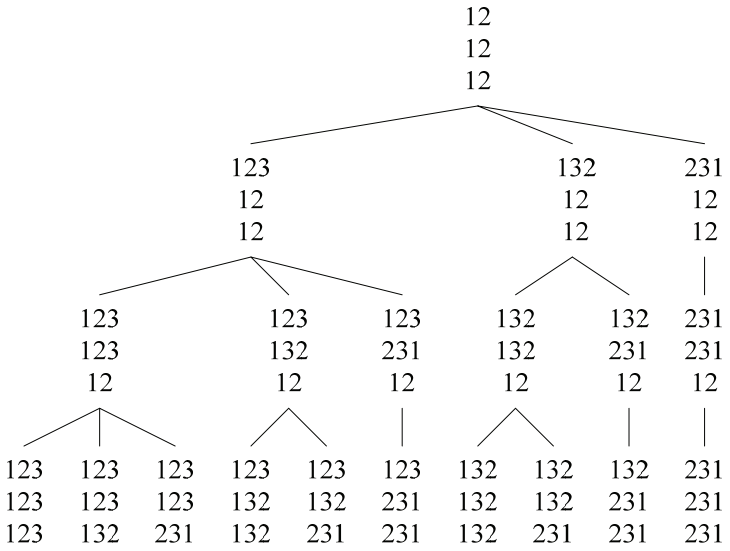
pakeitimo atžvilgiu sprendinių būtų ieškoma, medžio šaknis yra „12/12“. Taigi pirmąjį objektą vaizduojantis taškas niekada nebus dešiniau arba aukščiau už antrąjį objektą vaizduojantį tašką (žr. 3.1 pav.). Šio dalinio sprendinio įtempimo funkcijos rėžis nėra vertinamas, nes jis aprašo unikalių sprendinių aibę. Trečiąjį objektą vaizduojantis taškas horizontaliai gali būti įterpiamas dešiniau pirmųjų dviejų, tarp jų arba kairiau jų. Tokie įterpimai aprašomi trimis paieškos medžio šakomis, kurių viršūnės yra „123/12“, „132/12“ ir „231/12“. Vertikaliai trečiasis taškas gali būti įterptas aukščiau pirmųjų dviejų, tarp jų arba žemiau jų. Dėl to viršūnėje „123/12“ prasideda trys šakos, kurių viršūnės yra „123/123“, „123/132“ ir „123/231“.

Kad tik vieno iš invariantinių koordinačių sukeitimo atžvilgiu sprendinių būtų ieškoma, vienodo ilgio kėliniams yra apribojimai. Kėlinių tvarka laikoma tokia, kaip aprašyta ankstesniame skyrelyje. Kėlinys P_k negali būti ankstesnis už P_l, kai $k > l$. Dėl to daliniai sprendiniai „132/123“, „231/123“ ir „231/132“ nėra leidžiami, nes jie yra ekvivalentūs atitinkamai „123/132“, „123/231“ ir „132/231“ (žr. 3.2 pav.). Kai $d = 2$, unikalių sprendinių yra $m!^2 / 8 + m!/ 4$. Esant $d > 1$, kai kurios sprendinių aibės yra aprašomos daliniais sprendiniais su skirtingo ilgio kėliniais, pavyzdžiui, „123/12“. Tačiau dalinės įtempimo funkcijos reikšmės yra vertinamos tik daliniams sprendiniams su vienodo ilgio kėliniais.



3.4 pav. Šakų ir rėžių algoritmo DS paieškos medis, kai $d = 2$

Paieškos medis, kai $d = 3,$ pavaizduotas 3.5 paveiksle. Žymėjimai ir apribojimai yra panašūs į vartotus $d = 2$ atveju. Kai $d = 3,$ unikalių sprendinių skaičius lygus $m!^3 / 48 + m!^2 / 8 + m!/ 6.$ Panašiai gali būti sudaromi paieškos medžiai ir kai *d* didesnis.



3.5 pav. Šakų ir rėžių algoritmo DS paieškos medis, kai $d = 3$

Toliau pateikiame šakų ir rėžių algoritmą DS su miesto kvartalo atstumais. Algoritmo struktūra yra panaši į aprašytą [17] darbe. Naudojama paieškos gilyn strategija. Jos privalumas – nuosekliai peržiūrimos paieškos medžio viršūnės, dėl to nereikia saugoti neanalizuotų medžio viršūnių. Tai ypač aktualu, kai paieškos medis yra didelis. Paieškai platyn reikia naudoti eilės duomenų struktūrą kandidatams saugoti. Geriausiojo išrinkimo paieškai naudojama prioritetinė kandidatų eilė, kuriai reikia ne tik atminties resursų, bet įterpimas ir pašalinimas atliekamas per laiką, logaritmiškai priklausantį nuo kandidatų skaičiaus. Algoritme *j* yra didžiausia iš $\overline{m}$ kėlinių rinkinyje. Pagrindinis algoritmo ciklas kartojamas tol, kol $j > 2.$ Tai užtikrina, kad bus ieškoma tik vieno iš sprendinių, invariantinių koordinačių ašių krypties pakeitimo atžvilgiu. Algoritmas skiriasi nuo pateikto ankstesniame skyrelyje unikalių sprendinių perrinkimo algoritmo dalinio sprendinio įvertinimo sprendžiant kvadratinio programavimo uždavinį bloku, kuris atitinka rėžių skaičiavimą.

Šakų ir rėžių algoritmas daugiamatėms skalėms

Pradiniai duomenys: $m,\ d,\ \delta_{ij}, w_{ij}, i, j = 1,\ldots,m$

Rezultatas: $S^*,\ Y^*$

$p_{ki} \leftarrow i,\ i = 1,\ldots,m,\ k = 1,\ldots,d;\ \ j \leftarrow m+1;\ \ k \leftarrow d+1;\ \ S^* \leftarrow \infty$

Kol $j > 2$

Jei $k > d$

Jei $j > 2$ ir $j < m$

Jei $\min\limits_{\overline{Y} \in A(\overline{P})} \overline{S}(\overline{Y}) \geq S^*$ // Dalinio sprendinio įvertinimas

$k \leftarrow k-1$

Kitu atveju $j \leftarrow j+1;\ k \leftarrow 1$

Kitu atveju $j \leftarrow j+1;\ k \leftarrow 1$

Jei $j > m$

Jei $\min\limits_{Y \in A(P)} S(Y) < S^*$ // Sprendinio įvertinimas

$S^* \leftarrow \min\limits_{Y \in A(P)} S(Y);\ Y^* \leftarrow \arg\min\limits_{Y \in A(P)} S(Y)$

$j \leftarrow m;\ k \leftarrow d$

Jei $j > 2$ // Formuojamas kitas kėlinių rinkinys

Jei $p_{kj} = 0$

$p_{kj} \leftarrow j$

Jei $k > 1$ ir $P_k \prec P_{k-1}$ // Ekvivalentumo patikrinimas

$p_{ki} \leftarrow p_{(k-1)i}, i = 1,\ldots,j$

$k \leftarrow k+1$

Kitu atveju

$p_{kj} \leftarrow p_{kj} - 1$

Jei $p_{kj} = 0$

$p_{ki} \leftarrow p_{ki} - 1, i = 1,\ldots,j-1;\ k \leftarrow k-1$

Jei $k < 1$: $j \leftarrow j-1;\ k \leftarrow d$

Kitu atveju

$i \leftarrow j-1$ // Rasti i tokį, kad

Kol $p_{ki} \neq p_{kj}$: $i \leftarrow i-1$ // $p_{ki} = p_{kj}$

$p_{ki} \leftarrow p_{ki} + 1;\ k \leftarrow k+1$

Naudojantis tokiu šakų ir rėžių algoritmu, galima garantuotai išspręsti DS su miesto kvartalo atstumais uždavinius iki kelių tūkstančių kartų greičiau negu perrinkimo algoritmu. Dar svarbiau, kad didesnės apimties uždaviniai gali būti išspręsti per priimtiną laiką. Kai $d = 2$, šiuolaikišku asmeniniu kompiuteriu pagal šį algoritmą galima garantuotai išspręsti $m = 12$ uždavinį, o kai $d = 3$, – $m = 8$ uždavinį, kuriems reikia garantuotai išspręsti 24 kintamųjų globalaus minimizavimo uždavinį.

3.3.5. Kombinatorinis evoliucinis algoritmas

Evoliucinės paieškos idėja aprašyta 3.3.1 skyrelyje. Kai evoliucinė paieška naudojama viršutinio lygmens kombinatorinio uždavinio sprendimui dviejų lygmenų minimizavimo algoritme DS su miesto kvartalo atstumais, kitaip koduojami sprendiniai. Šiuo atveju kėlinių rinkinys P sudaro individams atstovaujančias chromosomas.

Populiacija vystosi generuojant dviejų atsitiktinių esamos populiacijos individų palikuonį, kurio chromosoma randama naudojantis formule

$$P_k = (\hat{p}_{k1},\ldots,\hat{p}_{k\beta},\tilde{p}_{k1},\ldots,\tilde{p}_{k(\gamma-\beta)},\hat{p}_{k\gamma},\ldots,\hat{p}_{km}),\ k = 1,\ldots,d, \qquad (3.28)$$

čia β ir γ yra du atsitiktiniai skaičiai iš $\{1,\ldots,m\}$, $\hat{P}_k$ – vieno atsitiktinio esamos populiacijos individo chromosoma, o $\tilde{p}_{ki}$ – skaičiai iš aibės $\{1,\ldots,m\}$, nepriklausantys aibei $\{\hat{p}_{k1},\ldots,\hat{p}_{k\beta},\hat{p}_{k\gamma},\ldots,\hat{p}_{km}\}$ ir surikiuoti kaip ir antro atsitiktinio esamos populiacijos individo chromosomoje.

Palikuonis pagerinamas naudojantis lokalia paieška, pagrįsta kvadratiniu programavimu. Palikuonio gerumas yra apibrėžiamas įtempimo funkcijos reikšme atitinkamo apatinio lygmens kvadratinio programavimo uždavinio minimumo taške.

Didesnės apimties DS uždavinius galima spręsti pagal lygiagretųjį algoritmą, kuris apima evoliucinį viršutinio lygmens kombinatorinio uždavinio sprendimą [143], [156]. Kiekviename procesoriuje naudojama atskira populiacija [18]. Kiekvienas procesorius vykdo tą patį evoliucinį algoritmą, tik su skirtingomis atsitiktinių skaičių sekomis, inicializuojant skirtingas, nuo procesoriaus numerio priklausančias, atsitiktinių skaičių generatorių sėklas.

Skirtingų procesorių rezultatai surenkami pabaigus skaičiavimus. Iš šio lygiagretaus algoritmo rezultatų galima akivaizdžiai matyti kelių procesorių panaudojimo privalumą. Algoritmu rastas geresnis Morzės kodų painiojimo uždavinio ($m = 36$) sprendinys, kai $d = 2$, negu skelbti, pavyzdžiui, [16] darbe.

3.4. Minkovskio atstumų įtaka vaizdams

Šis skyrelis skirtas palyginti DS, kuriose naudojami Euklido ir miesto kvartalo atstumai. Svarbiausias DS panaudojimo tikslas yra vizualizuoti daugiamačius duomenis, dėl to tikslinga palyginti gaunamus vaizdus. Kiekybinis įvertinimas naudojantis (3.4) santykine paklaida ne visada informatyvus, nes neatskleidžia vizualizavimo kokybės, be to, nelabai tinka sprendiniams, gautiems minimizuojant skirtingas įtempimo funkcijas, palyginti.

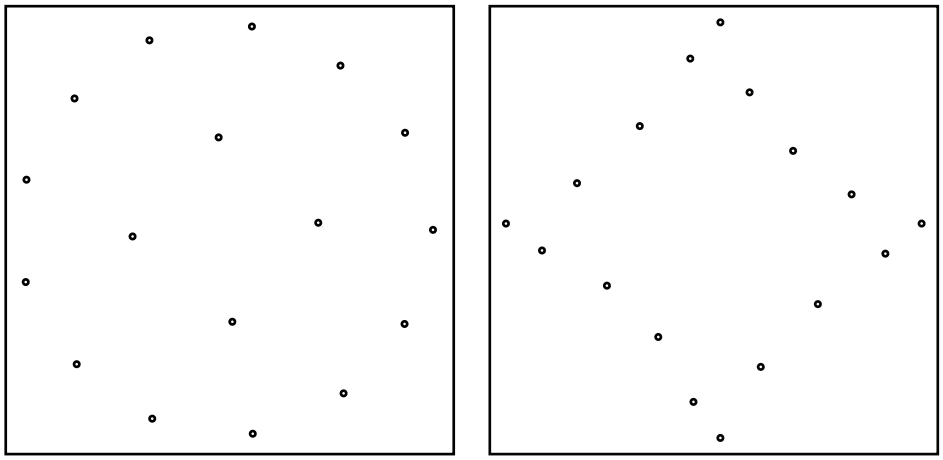
Kokybiniam palyginimui reikia tokios aibės duomenų, kurių vaizduose būtų galima ieškoti suprantamų daugiamačių objektų savybių. Vienas iš tokių yra gerai žinomoms geometrinėms figūroms priklausančių daugiamačių taškų aibės, pavyzdžiui, daugiamačių simpleksų ir kubų viršūnės. Nors vargu ar įmanoma įsivaizduoti geometrines figūras didesnio skaičiaus matmenų negu trimatėje erdvėje, tačiau jų savybės yra suprantamos. Pavyzdžiui, geometrinių figūrų simetriškumas lemia simetrines viršūnių pozicijas, dėl to tikimasi ir vaizdų simetrijos. Geometrinių figūrų ir praktinių duomenų aibių vaizdai DS, gauti naudojantis Euklido ir miesto kvartalo atstumais, kai $d = 2$, palyginti [154] darbe. Santykinės paklaidos reikšmės abiem atvejais yra panašios, tačiau miesto kvartalo atstumų jos truputį mažesnės. Lyginant geometrinių figūrų vaizdus, pirmenybė teikiama toms DS, kuriose išaiškėja numanomos daugiamačių geometrinių figūrų savybės. Skirtingo sudėtingumo minimizavimo uždaviniai gali būti sudaryti keičiant duomenų erdvės matmenų skaičių n ir tuo pačiu geometrinių figūrų viršūnių skaičių m.

Simpleksas yra iškilus n-matis daugiakampis, turintis $n+1$ viršūnių. Vienmatėje erdvėje simpleksas yra atkarpa, dvimatėje – trikampis, trimatėje – tetraedras, n-matėje erdvėje – tai daugiasienis, turintis kiek galima mažiausiai viršūnių. Taigi daugiamačio simplekso viršūnių skaičius yra $m = n+1$, o DS minimizavimo kintamųjų skaičius yra $(n+1)d$.

Gali būti naudojami kelių tipų simpleksai. Atstumai tarp standartinio simplekso viršūnių yra lygūs, dėl to skirtingumai $\delta_{ij} = 1, i \neq j$. Dvimatėje erdvėje toks simpleksas yra lygiakraštis trikampis, o trimatėje – taisyklingas tetraedras. Bet kokio skaičiaus matmenų erdvėje tokio simplekso savybė – visos viršūnės vienodai nutolusios nuo simplekso centro.

Standartinio 16-mačio simplekso viršūnių vaizdai DS, gauti naudojantis Euklido ir miesto kvartalo atstumais, pateikti 3.6 paveiksle. Čia 17 simplekso viršūnių yra pavaizduotos taškais. Miesto kvartalo atstumų atveju viršūnių vaizdai sudaro figūrą, panašią į kvadratą su vertikalia įstrižaine. Taigi jie yra

nutolę nuo centro panašiu atstumu, nes šių atstumų atveju kvadrato su vertikalia įstrižaine taškai yra vienodai nutolę nuo centro, panašiai kaip ir apskritimo taškai Euklido erdvėje. Kai DS skaičiuojami Euklido atstumai, viršūnių vaizdai sudaro apskritimus, tačiau skirtinguose apskritimuose esantys viršūnių vaizdai yra skirtingai nutolę nuo centro.



a) b)

3.6 pav. Standartinio 16-mačio simplekso viršūnių vaizdai DS, gauti naudojantis: a) Euklido atstumais, b) miesto kvartalo atstumais

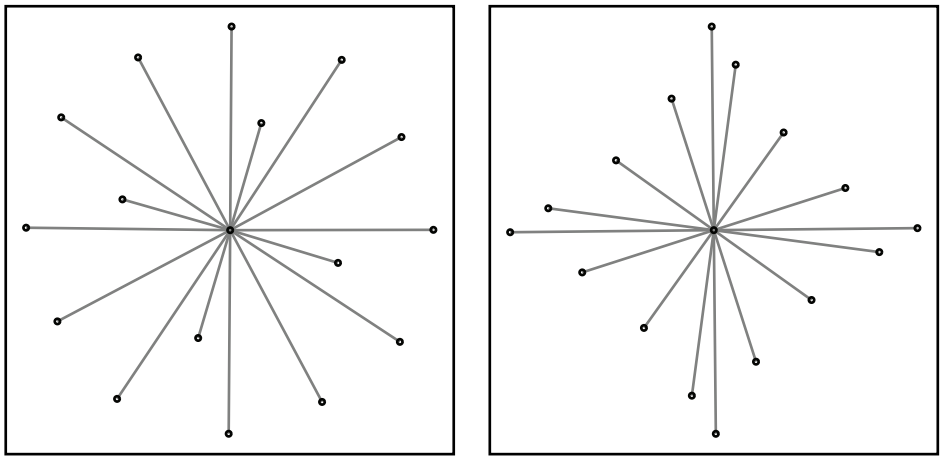
Vienetinio simplekso viena viršūnė yra koordinačių pradžioje, o kiekviena kita yra nutolusi kuria nors koordinačių kryptimi tokiu pat atstumu. Dvimatėje erdvėje toks simpleksas yra statusis lygiašonis trikampis. Bet kokio didesnio skaičiaus matmenų erdvėje tokio simplekso viršūnės yra dvejopos: viena viršūnė yra koordinačių pradžioje, o visos kitos yra nutolusios nuo pastarosios vienodu atstumu. Tokio simplekso viršūnių koordinatės

$$x_{ij} = \begin{cases} 1, & \text{jei } i = j+1, \\ 0 & \text{kitu atveju,} \end{cases}$$

čia $i = 1,\ldots,n+1$, $j = 1,\ldots,n$. Skirtingumai tarp viršūnių yra matuojami atstumu daugiamatėje duomenų erdvėje.

Kaip atrodo vienetinio 16-mačio simplekso viršūnių vaizdai DS, gauti naudojantis Euklido ir miesto kvartalo atstumais, matome 3.7 paveiksle. Čia 17 simplekso viršūnių pavaizduotos taškais. Linijos jungia viršūnę koordinačių

pradžioje vaizduojantį tašką su kitas viršūnes vaizduojančiais taškais. Koordinačių pradžioje esanti simplekso viršūnė pavaizduota centre. Miesto kvartalo atstumų atveju kitų viršūnių vaizdai sudaro figūrą, panašią į kvadratą su vertikalia įstrižaine. Taigi jie yra nutolę nuo viršūnės koordinačių pradžioje vaizdo panašiu atstumu. Vaizde DS, gautame naudojantis Euklido atstumais, kitų viršūnių vaizdai sudaro apskritimus, tačiau skirtinguose apskritimuose esantys viršūnių vaizdai yra skirtingai nutolę nuo viršūnės koordinačių pradžioje vaizdo.



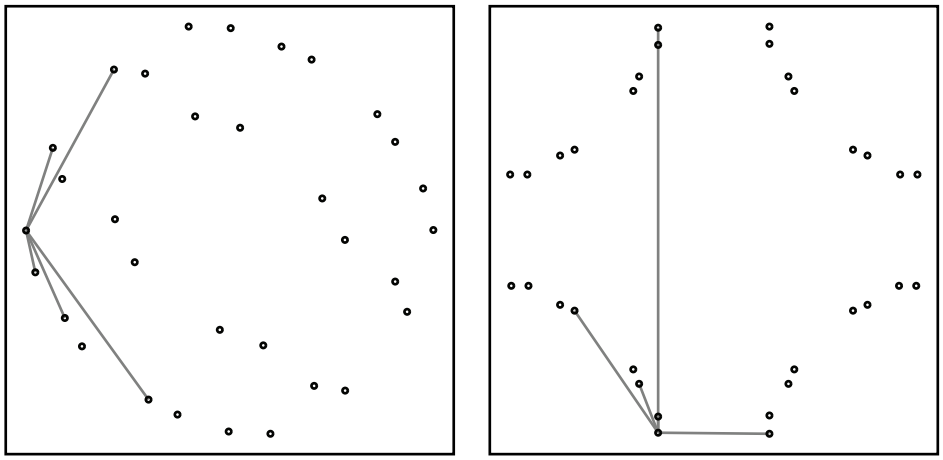
a) b)

3.7 pav. Vienetinio 16-mačio simplekso viršūnių vaizdai DS, gauti naudojantis: a) Euklido atstumais, b) miesto kvartalo atstumais

Daugiamatis kubas (vadinamasis hiperkubas) yra kvadrato ir įprasto kubo apibendrinimas daugiamatėje erdvėje. Daugiamačio kubo viršūnių skaičius – $m = 2^n$, o DS minimizavimo kintamųjų skaičius lygus $2^n d$. Vienas iš galimų daugiamačio kubo viršūnių koordinačių variantų yra toks: kubo viršūnės koordinatės yra lygios 0 arba 1, jas nustato dvejetainis viršūnės numerio $i = 0,\ldots,m-1$ kodas – vienas bitas vienai viršūnei. Skirtingumai tarp viršūnių yra matuojami atstumu daugiamatėje duomenų erdvėje. Kubo savybė – visos viršūnės vienodai nutolusios nuo kubo centro, jos sudaro viršūnių grupes, atitinkančias briaunas, šonus ir pan.

Kaip atrodo 5-mačio kubo viršūnių vaizdai DS, gauti naudojantis Euklido ir miesto kvartalo atstumais, parodyta 3.8 paveiksle. Čia 32 kubo viršūnės

pavaizduotos taškais. Linijos vaizduoja kai kurias briaunas – jungia vieną iš viršūnių vaizduojantį tašką ir gretimas jai viršūnes vaizduojančius taškus. Daugiamačio kubo viršūnių vaizdai DS, gauti naudojantis miesto kvartalo atstumais, sudaro figūrą, panašią į kvadratą su vertikalia įstrižaine. Taigi jie yra nutolę nuo centro panašiu atstumu. Vaizde DS, gautame naudojantis Euklido atstumais, tokia kubų savybė neišlieka. Abiejų atstumų atveju kubo viršūnių vaizdai suformuoja mažesnio skaičiaus matmenų kubus atitinkančias grupes – briaunų ir šonų vaizdus.



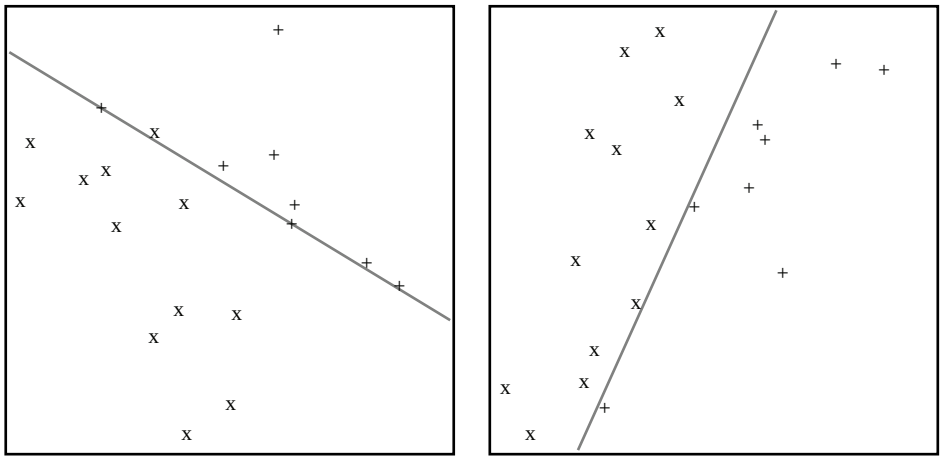
a) b)

3.8 pav. 5-mačio kubo viršūnių vaizdai DS, gauti naudojantis: a) Euklido atstumais, b) miesto kvartalo atstumais

Kitas tipas duomenų aibių, kurių vaizdai gali būti panaudoti skirtingų atstumų įtakos DS analizei, yra empiriniai duomenys, kurių savybės yra žinomos arba numanomos. Pirmenybė teikiama toms DS, kuriose yra matomos numanomos daugiamačių empirinių duomenų savybės. Pavyzdžiui, [159] darbe DS pritaikytos farmakologinio sąryšio duomenims vizualizuoti. Tokie duomenys pateikiami matrica. Vienas jos matmuo atitinka baltymus, o kitas – mažesniąsias molekules (vadinamuosius ligandus), kurios prisijungia prie baltymų. Ligandai yra natūralūs neurotransmiteriai ir vaistai, kurie, jungdamiesi prie baltymų, keičia jų funkcijas – blokuoja arba aktyvuoja. Analizuojant farmakologinio sąryšio duomenis kaip ligandų parametrus, galima tikėtis, kad aktyvuojančiuosius ligandus bus galima atskirti nuo blokuojančiųjų.

Farmakologinio sąryšio duomenų vaizdai DS, gauti naudojantis Euklido ir miesto kvartalo atstumais, kai $d = 2$, palyginti [159] darbe. Nors abiem atvejais DS vaizduose matyti panašios duomenų savybės, tačiau kai kurios jų labiau išryškėjusios, kai naudojami miesto kvartalo atstumai. Be to, šiuo atveju (3.4) santykinės paklaidos reikšmės yra mažesnės. Farmakologinių duomenų vizuali analizė detaliau nagrinėjama 5.2.4 skyrelyje, šiame skyriuje aptarsime tik vienos iš farmakologinių duomenų aibės vizualizavimą.

Farmakologinių duomenų, nagrinėtų [126] darbe, vaizdai DS, gauti vizualizuojant ligandų sąryšio su baltymais parametrus naudojantis Euklido ir miesto kvartalo atstumais, pateikti 3.9 paveiksle. Aktyvuojančiosios molekulės pavaizduotos ženklu „+“, o blokuojančiosios – ženklu „x“.



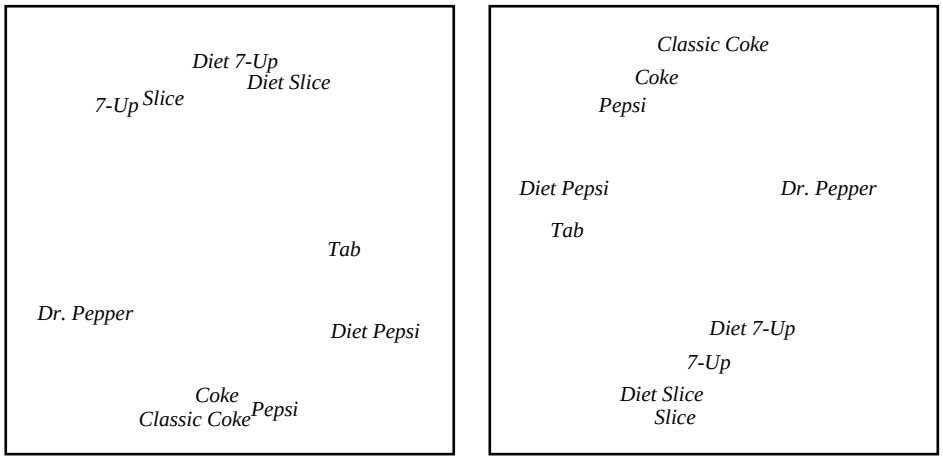
a) b)

3.9 pav. Farmakologinių duomenų vaizdai DS, gauti naudojantis: a) Euklido atstumais, b) miesto kvartalo atstumais

Vaizde DS, gautame naudojantis miesto kvartalo atstumais, yra įmanoma nubrėžti aktyvuojančiųjų ir blokuojančiųjų ligandų vaizdus atskiriančią tiesę. Tačiau vaizde DS, gautame naudojantis Euklido atstumais, to padaryti nepavyksta. Dėl to galima teigti, kad naudojantis miesto kvartalo atstumais ligandų atskyrimas į blokuojančiuosius ir aktyvuojančiuosius yra išryškėjęs labiau negu naudojantis Euklido atstumais.

Dažnai naudojamas vaizdus DS demonstravimo pavyzdys yra pagrįstas eksperimentinio gaiviųjų gėrimų testavimo rezultatais [56]. Eksperimente

dalyvavo 38 studentai, kurie ragavo 10 gaiviųjų gėrimų. Kiekviena gėrimų pora buvo įvertinta jų skirtingumu 9 balų skalėje (1 – labai panašūs, 9 – visiškai skirtingi). Daugiamačių duomenų aibę sudaro susumuoti skirtingumo įvertinimai. Šio uždavinio tikslas – mažo matmenų skaičiaus erdvėje rasti 10 gėrimų vaizduojančių taškų išsidėstymą, kuris padėtų interpretuoti duomenis. Gaiviųjų gėrimų vaizdai DS pateikti 3.10 paveiksle. Vaizdai, gauti naudojantis Euklido atstumais, suformuoja į apskritimą panašią figūrą, o naudojantis miesto kvartalo atstumais – pasuktą stačiakampį.



3.10 pav. Gaiviųjų gėrimų vaizdai DS, gauti naudojantis: a) Euklido atstumais, b) miesto kvartalo atstumais

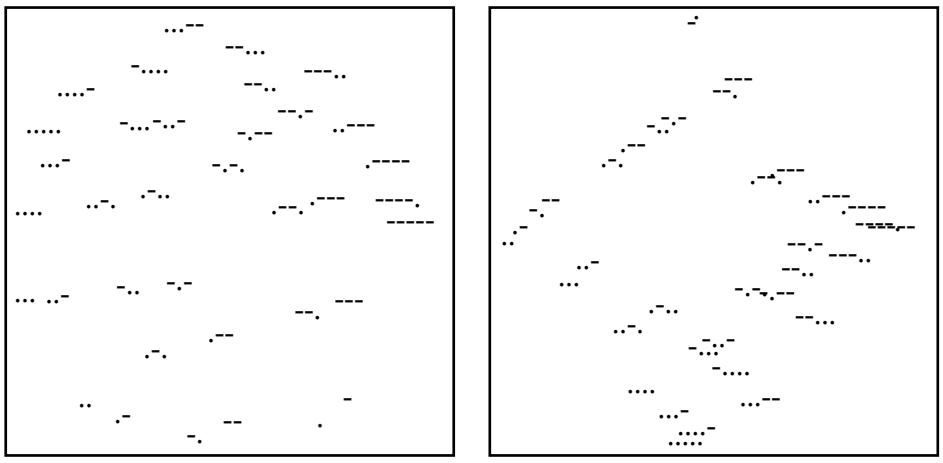
Abiejuose vaizduose išsiskiria gaiviųjų gėrimų grupės. Vieną ryškią grupę sudaro gėrimai *Coke*, *Classic Coke* ir *Pepsi*, antrąją – *Slice* ir *7-Up* bei jų dietinės versijos. Gėrimas *Dr. Pepper* ganėtinai skiriasi nuo kitų. Iš vaizdo DS, gauto naudojantis Euklido atstumais, matyti, kad dietiniai gėrimai ir gėrimas *Tab* yra atsiskyrę. Vaizde DS, gautame naudojantis miesto kvartalo atstumais, *Slice* ir *7-Up* dietiniai gėrimai yra nelabai nutolę nuo klasikinių, tačiau gautame naudojantis Euklido atstumais atrodo kitaip – *7-Up* yra arčiau *Slice* negu *Diet 7-Up*. Taigi šiuo atveju gaunami skirtingi vaizdai, bet sunku įvertinti, kuriems turi būti teikiamas privalumas. Dėl to naudinga peržiūrėti visus galimus vaizdus, tai gali suteikti daugiau informacijos apie analizuojamus duomenis.

Dar vienas empirinis DS uždavinys yra Morzės kodų painiojimo duomenys [14]. Šiuos duomenis sudaro $m = 36$ lotyniškų raidžių ir skaitmenų

Morzės kodų skirtingumai, surinkti psichologinio eksperimento metu. Respondentų buvo prašoma įvertinti, ar du vienas po kito garsiniu signalu išgauti kodai yra vienodi.

Raidžių ir skaitmenų Morzės kodą sudaro seka iki penkių simbolių – taškų ir brūkšnių, kurie garsiniu būdu išreiškiami kaip trumpas (0,05 sekundės) ir ilgas (0,15 sekundės) pyptelėjimai, atskirti trumpa (0,05 sekundės) pauze. Pavyzdžiui, raidės „S“ kodas yra „...“, o raidės „O“ kodas yra „---“. Tokių duomenų analizė gali parodyti, į ką reikia atkreipti dėmesį mokinantis Morzės kodus ir kaip patikrinti tokias žinias.

Morzės kodų painiojimo duomenų vaizdai DS, gauti naudojantis Euklido ir miesto kvartalo atstumais, pateikti 3.11 paveiksle. Iš šių vaizdų matyti, kad labiausiai painiojami kodai, kuriuose skiriasi tik paskutinis simbolis.



a) b)

3.11 pav. Morzės kodų painiojimo duomenų vaizdai DS, gauti naudojantis: a) Euklido atstumais, b) miesto kvartalo atstumais

Kodai iš vieno simbolio gana lengvai atskiriami nuo kitų. Yra matomi kodų iš vieno, dviejų ir trijų simbolių vaizdų klasteriai, o kodų iš keturių ir penkių simbolių vaizdai susigrupuoja gana glaustai. Kodų iš dviejų simbolių vaizdai glausčiau susigrupavę DS, kai naudojamasi miesto kvartalo atstumais, tačiau tokiu atveju kodų iš trijų simbolių vaizdai sudaro dvi nutolusias grupes.

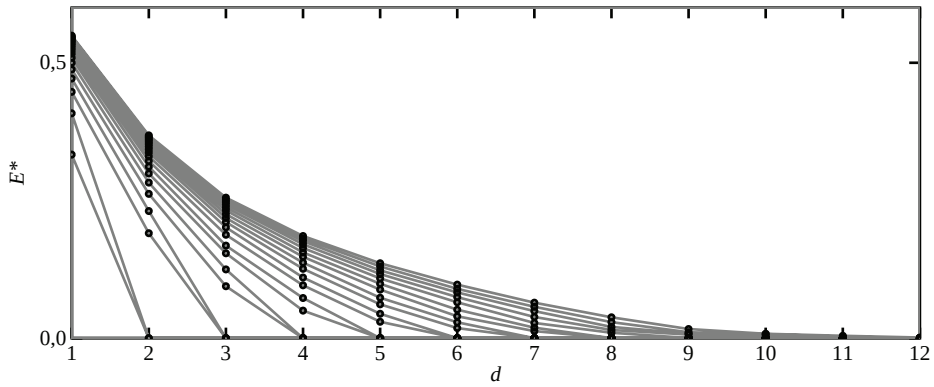
Dažniausiai naudinga peržiūrėti vaizdus DS, gautus naudojantis įvairiais Minkovskio atstumais, nes jie gali suteikti daugiau informacijos apie

analizuojamus daugiamačius duomenis negu stebint vaizdus vieno tipo DS. Kaip matyti iš pateiktų pavyzdžių, DS gali labiau atsispindėti kai kurios objektų savybės, kai naudojamasi miesto kvartalo atstumais. Tačiau 3.2 skyrelyje analizuodami įtempimo funkcijos diferencijuojamumą, matėme, kad tokiu atveju DS uždaviniai yra sunkiau sprendžiami.

3.5. Skalių matmenų skaičiaus įtaka daugiamačių duomenų analizei

Trimačio vizualizavimo taikymams pagrįsti panagrinėkime (3.4) santykinės paklaidos priklausomybę nuo vaizdo erdvės matmenų skaičiaus d. Eksperimentiškai nustatant geometrinių figūrų ir empirinių duomenų priklausomybę buvo naudojamas DS su miesto kvartalo atstumais algoritmas.

Daugiamačių simpleksų santykinės DS paklaidos priklausomybė nuo vaizdo erdvės matmenų skaičiaus d parodyta 3.12 paveiksle. Skirtingos kreivės vaizduoja skirtingo matmenų skaičiaus daugiamačių simpleksų santykinės paklaidos priklausomybę. Kaip buvo galima tikėtis, santykinės paklaidos mažėja didinant vaizdo erdvės matmenų skaičių. Iš paveikslo matyti, kad paklaida pastebimai sumažėja ne tik pereinant nuo vienmatės skalės prie dvimatės, bet ir nuo dvimatės prie trimatės.

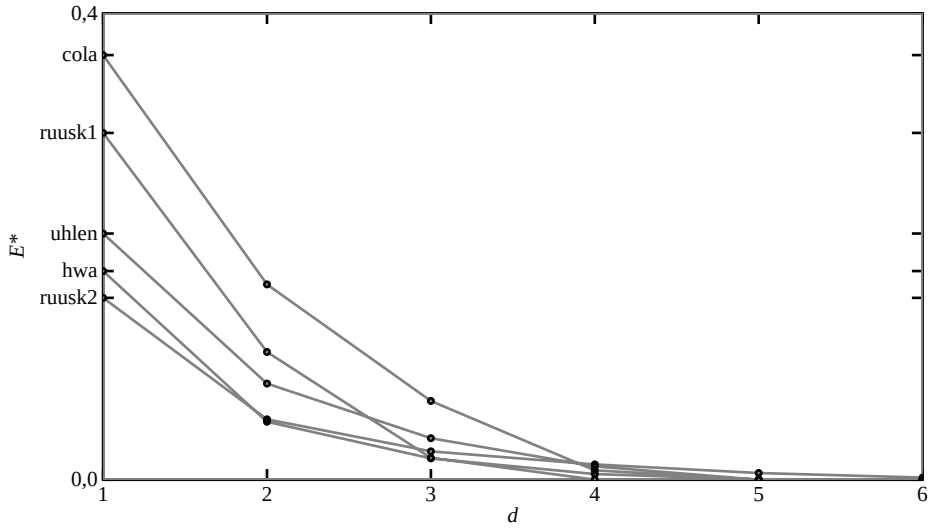


3.12 pav. Daugiamačių simpleksų santykinės DS paklaidos priklausomybė nuo vaizdo erdvės matmenų skaičiaus d

Nagrinėtos empirinės daugiamačių duomenų aibės yra gaiviųjų gėrimų testavimo duomenys [56] „cola“ ir farmakologinio sąryšio duomenys [159]: „ruusk1“ žymi farmakologinio sąryšio duomenis iš [126], analizuojamus kaip trijų žmogaus ir penkių žuvytės zebrinės danijos (*Danio rerio*) α_2-

adrenoceptorinių baltymų ($m = 8$) parametrai; „ruusk2“ – tie patys duomenys, analizuojami kaip 20 su šiais baltymais susijungiančių ligandų ($m = 20$) parametrai; „uhlen“ – farmakologinio sąryšio duomenys iš [140], analizuojami kaip žmogaus, žiurkės, jūrų kiaulytės ir kiaulės α_2-adrenoceptorinių baltymų ($m = 12$) parametrai; „hwa“ – farmakologinio sąryšio duomenys iš [76], analizuojami kaip natūralių ir mutuotų α_1-adrenoceptorinių baltymų ($m = 12$) parametrai.

Šių duomenų santykinės DS paklaidos priklausomybė nuo vaizdo erdvės matmenų skaičiaus d pavaizduota 3.13 paveiksle. Skirtingos kreivės vaizduoja skirtingų aibių priklausomybę. Kaip ir tikėtasi, santykinės paklaidos mažėja didinant vaizdo erdvės matmenų skaičių d. Be to, šių duomenų aibių santykinės paklaidos mažėja kur kas greičiau negu daugiamačių simpleksų viršūnių aibių. Kaip ir simpleksų viršūnių aibių atveju, paklaida pastebimai mažėja ne tik pereinant nuo vienmatės skalės prie dvimatės, bet ir nuo dvimatės prie trimatės. Tai svarbu norint pagrįsti erdvinio vizualizavimo priemonių panaudojimą.



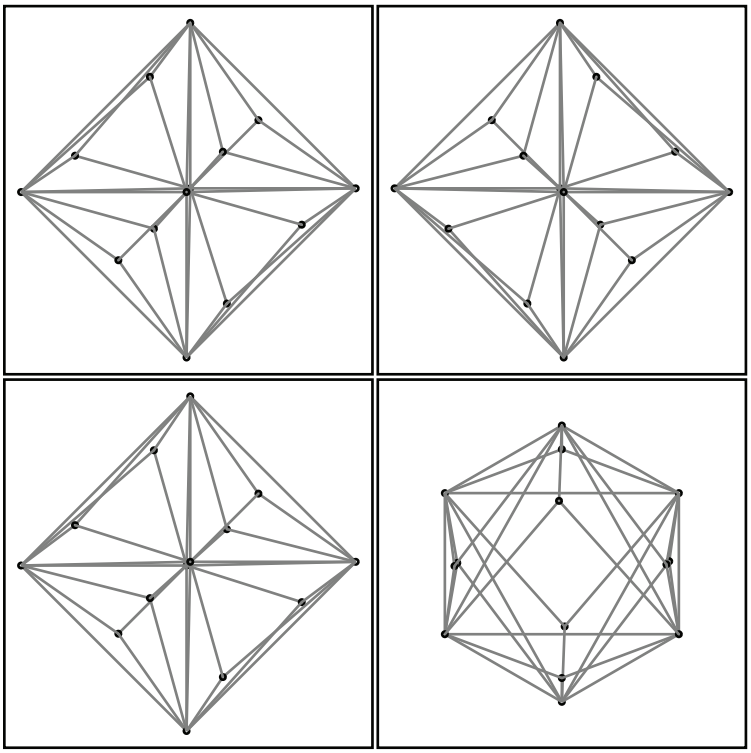
3.13 pav. Empirinių daugiamačių duomenų santykinės DS paklaidos priklausomybė nuo vaizdo erdvės matmenų skaičiaus d

Didžiausios paklaidos yra nagrinėjant „cola“ duomenų aibę, nes šiuo atveju skirtingumo duomenys gauti apklausus respondentus ir yra subjektyvūs. Kitų aibių skirtingumai yra atstumai tarp daugiamačių taškų, kurių realus matmenų

skaičius nedidėja taip greitai kaip objektų skaičius. Tinkamiausia empirinėms aibėms yra trimatė vaizdo erdvė. Trimatėse skalėse turėtų išlikti daugiau duomenų savybių negu dvimatėse. Vis populiaresni tampa erdviniai ekranai ir kitos erdvinio vizualizavimo priemonės, todėl ir trimatės skalės ypač patrauklios. Tačiau pademonstruoti jų privalumus popieriuje ar paprastame monitoriuje gana sunku.

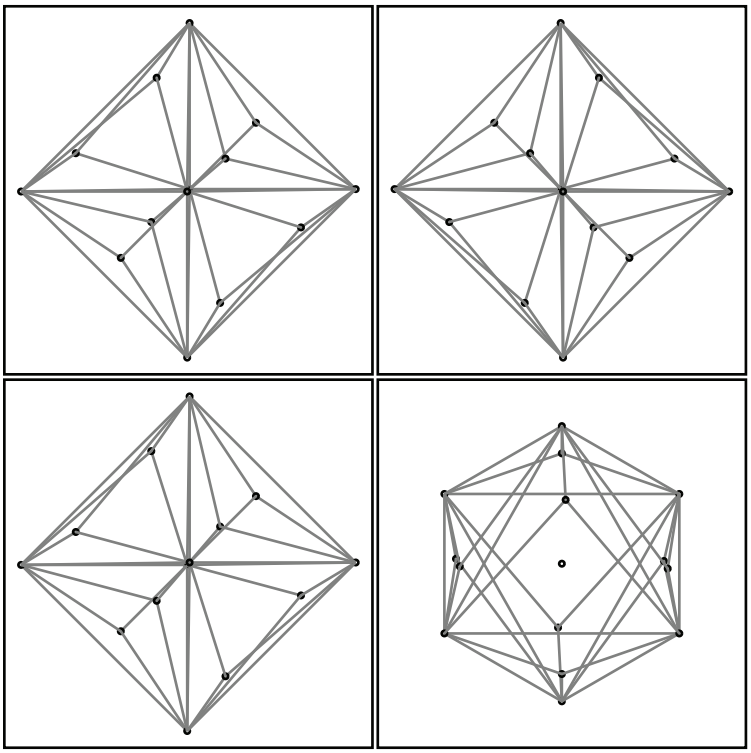
Klasikinis erdvinių kūnų vaizdavimo ortogonaliomis ir izometrine projekcijomis būdas yra ne toks vaizdus kaip naudojant erdvinius ekranus, tačiau net ir iš projekcijų galima įvertinti kai kurias erdvinių objektų savybes, ką galima panaudoti analizuojant trimates skales.

Daugiamačių geometrinių figūrų vaizdai trimatėse skalėse pavaizduoti ortogonaliomis bei izometrine projekcijomis ir aptarti [155] darbe. Standartinio 13-mačio simplekso viršūnių vaizdo trimatėje skalėje keturios projekcijos pateiktos 3.14 paveiksle.



3.14 pav. Standartinio 13-mačio simplekso viršūnių vaizdo trimatėje skalėje projekcijos

Panašiai 3.15 paveiksle pateiktos vienetinio 14-mačio simplekso viršūnių vaizdo trimatėje skalėje projekcijos. Vaizdai skiriasi tik tuo, kad šiuo atveju viršūnių yra viena daugiau.



3.15 pav. Vienetinio 14-mačio simplekso viršūnių vaizdo trimatėje skalėje projekcijos

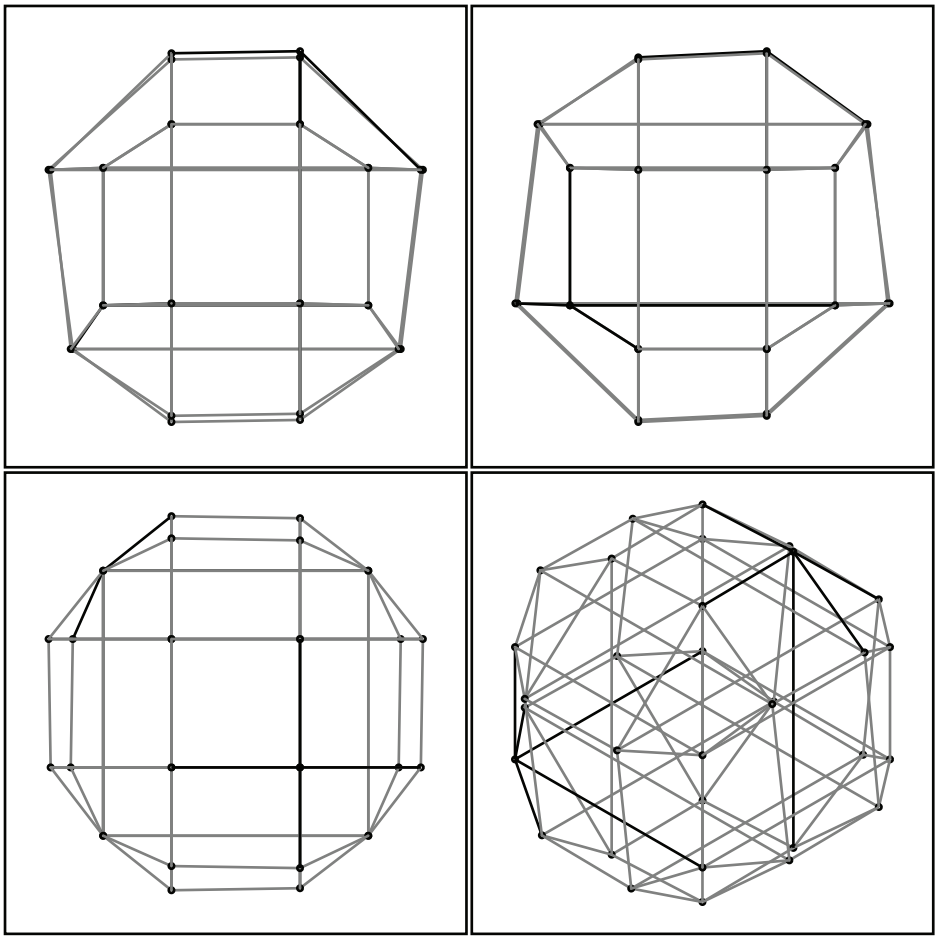
Ortogonali projekcija plokštumoje *xz* pavaizduota kairėje viršuje, ortogonali projekcija plokštumoje *yz* – dešinėje viršuje, ortogonali projekcija plokštumoje *xy* – kairėje apačioje, o izomerinė projekcija, kurioje visų trijų ašių masteliai yra vienodi, tik kampas tarp bet kurių dviejų ašių yra 120 laipsnių, – dešinėje apačioje. Ortogonaliose projekcijose ašis *x* yra nukreipta į kairę, ašis *z* – į viršų, o ašis *y* dešiniajame viršutiniame paveiksle nukreipta į dešinę ir apatiniame kairiajame paveiksle – žemyn. Izometrinėje projekcijoje ašis *x* nukreipta kairėn žemyn, ašis *y* – dešinėn žemyn, o ašis *z* – į viršų.

Simplekso viršūnės yra pavaizduotos taškais. Standartinio simplekso viršūnių vaizdai yra nutolę nuo figūros centro panašiu atstumu. Vienetinio

simplekso viršūnė koordinačių pradžioje atvaizduota figūros centre, kitų viršūnių vaizdai nutolę nuo jos panašiu atstumu

Linijos vaizduoja gautų trimačių figūrų briaunas – figūros išorėje esančius sujungimus tarp viršūnių. Trimatė figūra yra labai panaši į sferos atitikmenį miesto kvartalo atstumų atveju, tik viršūnių vaizdai, nesantys sferos atitikmens kampuose, yra šiek tiek išsikišę iš jos šonų.

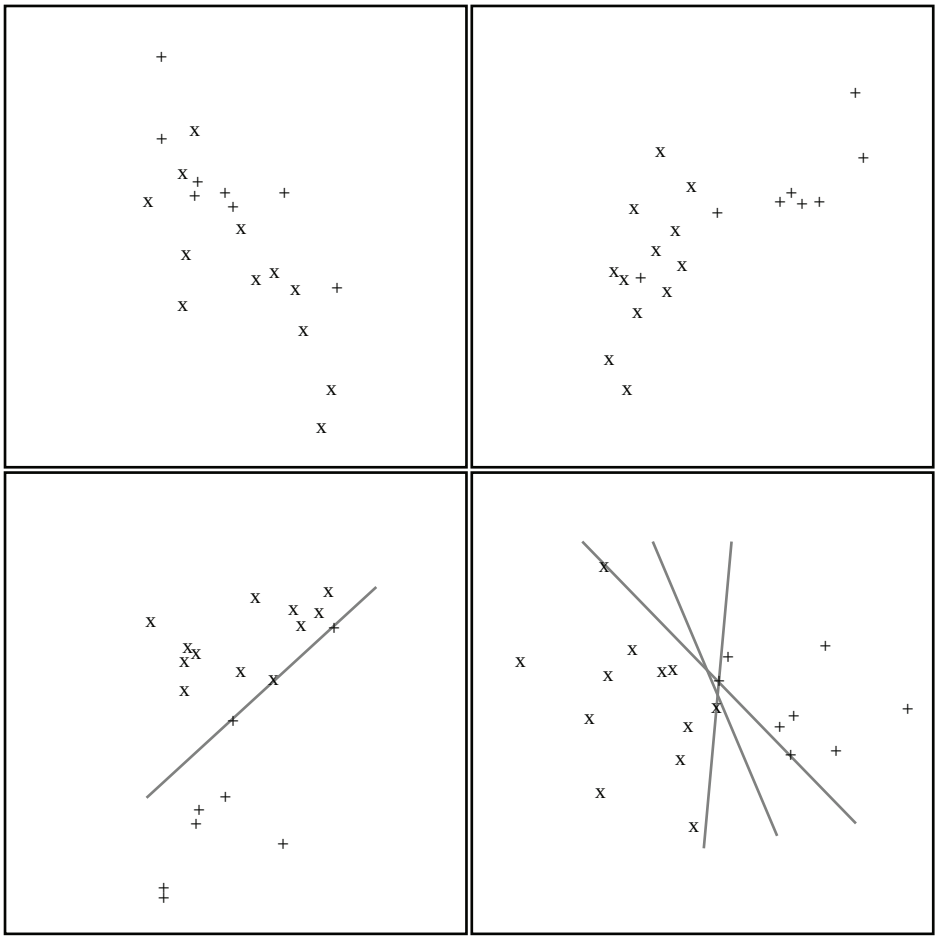
3.16 paveiksle matome 5-mačio kubo viršūnių vaizdo trimatėje skalėje projekcijas. Linijos vaizduoja daugiamačio kubo briaunas. Ortogonaliose projekcijose dauguma viršūnių vaizdų vienas kitą užstoja ir sunku įsivaizduoti figūros tūrį, tačiau tai yra lengviau matant izometrinę projekciją.



3.16 pav. 5-mačio kubo viršūnių vaizdo trimatėje skalėje projekcijos

Farmakologinio sąryšio duomenų „ruusk2“ vaizdų trimatėje skalėje projekcijas matome 3.17 paveiksle. Aktyvuojančiosios molekulės pavaizduotos ženklais „+“, o blokuojančiosios – „x“. Ši duomenų aibė tyrinėta naudojantis dvinarių medžių klasterizavimu ir pagrindinių komponenčių analize [126] darbe.

Gautuose dvinariuose medžiuose aktyvuojančiosios molekulės sudaro vieną grupę, išskyrus vieną molekulę, kuri yra blokuojančiųjų grupėje. Trijų pagrindinių komponenčių vaizde visos aktyvuojančiosios molekulės yra vienoje grupėje, tačiau viena blokuojančioji molekulė yra arti aktyvuojančiųjų grupės.



3.17 pav. Farmakologinių duomenų vaizdų trimatėje skalėje projekcijos

Panašiai dvimatės skalės vaizde, gautame naudojantis Euklido atstumais, viena blokuojančioji molekulė trukdo tiese atskirti aktyvuojančiųjų ir blokuojančiųjų molekulių grupes (žr. 3.4 skyrelį). Tačiau dvimatės skalės vaizde, gautame naudojantis miesto kvartalo atstumais, galima nubrėžti aktyvuojančiųjų ir blokuojančiųjų molekulių grupes atskiriančią tiesę.

Ortogonali projekcija, pavaizduota 3.17 paveikslo kairiajame apatiniame kampe, yra gana panaši į veidrodinį dvimatės skalės vaizdą, pateiktą 3.9 paveiksle. Nors šioje projekcijoje dvi aktyvuojančiosios molekulės yra šalia blokuojančiųjų grupės, galima nubrėžti aktyvuojančiųjų ir blokuojančiųjų molekulių grupes atskiriančią tiesę.

Aktyvuojančiąją ir blokuojančiąją grupes lengviau atskirti nagrinėjant trimatį vaizdą. Nors tai gana sunku pademonstruoti popieriuje, bet izometrinėje projekcijoje, kaip matyti 3.17 paveikslo dešiniajame apatiniame kampe, galima nubrėžti įvairias skiriamąsias tieses. Taigi nagrinėjant trimatį vaizdą galima sudaryti įvairias skiriamąsias plokštumas.

Trimačius vaizdus lengviau analizuoti vizualizavus tam tikromis priemonėmis arba bent sukant kompiuterio ekrane. Tinklapyje www.mii.lt/enoc pateikti besisukantys erdviniai vaizdai, kurie padeda lengviau suvokti daugiamačių skalių metodu gautus trimačius vaizdus. Charakteringos trimačių vaizdų iliustracijos:

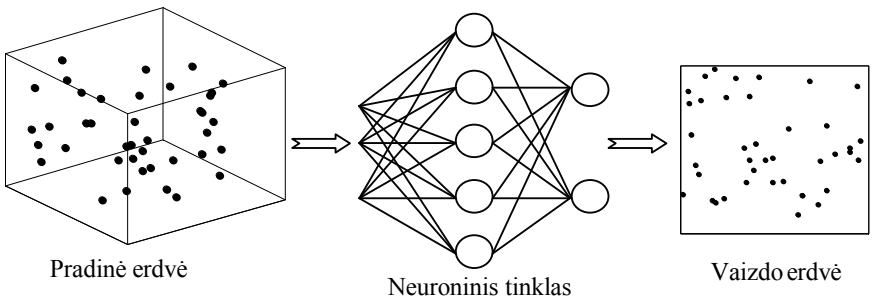
- standartinių simpleksų vaizdai;
- vienetinių simpleksų vaizdai;
- kubų vaizdai;
- trijų žmogaus ir penkių žuvytės zebrinės danijos baltymų parametrų vaizdai;
- 20 su žmogaus ir žuvytės zebrinės danijos baltymais susijungiančių ligandų parametrų vaizdai;
- žmogaus, žiurkės, jūrų kiaulytės ir kiaulės baltymų parametrų vaizdai;
- natūralių ir mutuotų baltymų parametrų vaizdai.

4. Dirbtiniai neuroniniai tinklai duomenims vizualizuoti

Dirbtiniai neuroniniai tinklai (DNT, *artificial neural networks*, ANN) yra taikomi ne tik klasifikavimo, klasterizavimo, funkcijų aproksimavimo, prognozavimo, optimizavimo uždaviniams spręsti, bet ir daugiamačių duomenų (objektų) matmenų skaičiui mažinti, tuo pačiu ir daugiamačiams duomenims (objektams) vizualizuoti. Dirbtiniai neuroniniai tinklai dažnai padeda atskleisti daugiamačių duomenų savybes, kurių negalima pastebėti klasikiniais daugiamačių duomenų vizualizavimo metodais.

Šiuo metu yra sukurta įvairių dirbtinių neuroninių tinklų ir jų mokymo algoritmų daugiamačiams duomenims vizualizuoti [88], [90], [106], [118], [129], [142]. DNT plataus spektro taikymams vizualizavimo procese skirtas [46] darbas. Neuroninio tinklo, skirto daugiamačių duomenų matmenų skaičiui mažinti, bendra schema pavaizduota 4.1 paveiksle.

Šiame skyriuje pateikiami dirbtinių neuroninių tinklų pagrindai, kurie būtini analizuojant DNT galimybes vizualizuoti daugiamačius duomenis. Čia supažindinama su biologiniu neuronu, pateikiamas dirbtinio neurono modelis, analizuojami neuroninių tinklų mokymo būdai, nagrinėjama vienasluoksnių ir daugiasluoksnių tiesioginio sklidimo neuroninių tinklų sandara, pateikiami jų mokymo algoritmai. Nagrinėjami dažniausiai naudojami dirbtiniai neuroniniai tinklai, skirti daugiamačiams duomenims vizualizuoti: saviorganizuojantys neuroniniai tinklai (SOM), SOM ir Sammono algoritmo junginiai, kreivinių komponenčių analizės metodas, daugiamačių skalių tipo projekcijos taikant dirbtinius neuroninius tinklus, autoasociatyvieji neuroniniai tinklai, neuroninių skalių metodas.



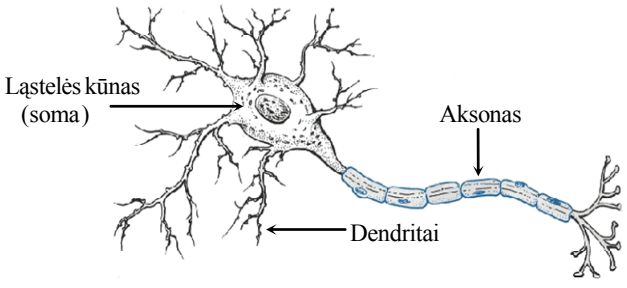
4.1 pav. Neuroninio tinklo daugiamačiams duomenims vizualizuoti schema

4.1. Dirbtinių neuroninių tinklų pagrindai

Dirbtiniai neuroniniai tinklai pradėti tyrinėti kaip biologinių neuroninių sistemų modelis. Pagrindinis dirbtinių neuroninių tinklų teorijos tikslas yra ne kuo detaliau modeliuoti biologinius neuronus, o išsiaiškinti ir pritaikyti biologinių neuronų sąveikos mechanizmus efektyvesnėms informacijos apdorojimo sistemoms kurti. Ateityje šis neuro biologinis modeliavimas gali padėti sukurti „protingus“ kompiuterius. Kai aišku iš konteksto, dirbtiniai neuroniniai tinklai vadinami tiesiog neuroniniais tinklais. Šiuo metu DNT vis plačiau naudojami dėl kelių pagrindinių priežasčių. Visų pirma, neuroninis tinklas yra galingas modeliavimo aparatas. Juo galima modeliuoti ypač sudėtingas funkcijas. Antra, neuroniniai tinklai turi galimybę mokytis iš pavyzdžių. Turint duomenų pavyzdžius ir naudojant mokymo algoritmus, neuroninis tinklas pritaikomas prie duomenų struktūros ir išmoksta atpažinti naujus duomenis, kurie nebuvo naudojami tinklo mokyme. DNT yra taikomi klasifikavimui, klasterizavimui, funkcijų aproksimavimui, prognozavimui, optimizavimui ir kt. Be daugelio kitų dirbtiniais neuroniniais tinklais sprendžiamų uždavinių, juos sėkmingai galima taikyti ir daugiamačiams duomenims vizualizuoti.

4.1.1. Biologinis neuronas

Žmogaus smegenys susideda iš daugelio (apie 10 000 000 000 000) neuronų, sujungtų vienų su kitais. Kiekvienas *neuronas* turi vidutiniškai keletą tūkstančių jungčių. Tai ląstelė, galinti generuoti elektrocheminį signalą. Neuronas turi išsišakojusią įėjimo struktūrą, vadinamuosius *dendritus*, ląstelės kūną, vadinamąją *somą*, ir besišakojančią išėjimo struktūrą – *aksoną* (žr. 4.2 pav.).



4.2 pav. Biologinis neuronas

Vienos ląstelės aksonas su kitos ląstelės dendritais jungiasi per *sinapses*. Kai sužadinama pakankamai neuronų, prijungtų prie neurono dendritų, tas neuronas taip pat sužadinamas ir generuoja elektrocheminį impulsą. Signalas per sinapses perduodamas kitiems neuronams, kurie vėl gali būti sužadinami. Neuronas sužadinamas tik tuo atveju, jei bendras dendritais gautas signalas viršija tam tikrą lygį, vadinamąjį *sužadinimo slenkstį*. Turint didžiulį skaičių visiškai paprastų elementų, kurių kiekvienas skaičiuoja svorinę įeinančių signalų sumą ir generuoja binarųjį signalą, jei suminis signalas viršija tam tikrą lygį, galima atlikti gana sudėtingas užduotis [144].

4.1.2. Dirbtinio neurono modelis

Praėjusio amžiaus ketvirtajame dešimtmetyje buvo pasiūlytas dirbtinio neurono modelis. Tai elementarus procesorius, skaičiuojantis įėjimo kintamojo netiesinę funkciją.

Pirmasis formalus dirbtinio neurono apibrėžimas pasiūlytas [110] darbe.

1. Neuronas turi keletą įėjimų. Kiekviena įėjimo x_k, $k = 1,\ldots,n$, jungtis turi savo perdavimo koeficientą (svorį) w_k, $k = 1,\ldots,n$ (žr. 4.3 pav.). Įprastai įėjimų x_k ir jungčių svorių w_k reikšmės yra realieji skaičiai.
2. Skaičiuojama įėjimo reikšmių ir svorių sandaugų suma

$$a = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n = \sum_{k=1}^{n} w_k x_k ,$$

kuri vadinama sužadinimo signalu.

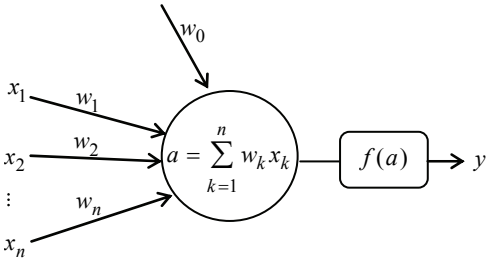
3. Neuroną apibūdina aktyvacijos funkcija

$$y = f(a) = f\left(\sum_{k=1}^{n} w_k x_k \right), \tag{4.1}$$

kurios reikšmė

$$f(a) = \begin{cases} 1, & \text{jei } a \geq w_0, \\ 0, & \text{jei } a < w_0, \end{cases} \tag{4.2}$$

vadinama neurono išėjimo reikšme, čia w_0 yra slenksčio reikšmė. Funkcijos $f(a)$ reikšmė lygi vienetui, kai sužadinimo signalas yra ne mažesnis už slenksčio reikšmę w_0, ir nuliui – priešingu atveju. Funkcijos $f(a)$ šuolis yra taške w_0, anglų kalboje jis vadinamas *bias*.



4.3 pav. Dirbtinis neuronas

Dažnai yra naudojamas sužadinimo signalas a, kuris aktyvuoja neuroną, kai $a = 0$:

$$a = \sum_{k=0}^{n} w_k x_k \ .$$

Šiuo atveju įvedamas nulinis įėjimas x_0, kuris yra pastovus: $x_0 = 1$.

Dirbtinio neurono modelyje gali būti naudojama ne tik slenkstinė funkcija (4.2), bet ir kitos neurono aktyvacijos funkcijos, pavyzdžiui, sigmoidinė funkcija

$$f(a) = \frac{1}{1 + \mathrm{e}^{-a}} ,$$

hiperbolinis tangentas

$$f(a) = \tanh(a) = \frac{\mathrm{e}^{a} - \mathrm{e}^{-a}}{\mathrm{e}^{a} + \mathrm{e}^{-a}}$$

ir kt.

Kaip matyti iš 4.2 ir 4.3 paveikslų, biologinis neuronas ir dirbtinio neurono modelis yra gana panašūs.

Dirbtiniai neuronai yra jungiami į *dirbtinius neuroninius tinklus*. Neuroninis tinklas gali būti pavaizduotas kaip grafas, kurio viršūnės yra neuronai, o lankai (su svoriais) jungia neuronų išėjimus su kitų, o kartais ir tų pačių, neuronų įėjimais [79].

Pagal jungimo konstrukciją neuroniniai tinklai sudaro dvi pagrindines grupes:

1) tiesioginio sklidimo (*feedforward*) tinklai, kuriuose nėra grafo ciklų;
2) grįžtamojo ryšio (*feedback*) tinklai, kuriuose yra grafo ciklai.

4.1.3. Dirbtinių neuroninių tinklų mokymas

Galimybė mokytis yra esminė intelekto savybė. Nors tikslų mokymo apibrėžimą sunku suformuluoti, dirbtinio neuroninio tinklo mokymo procesas apibrėžiamas kaip tinklo struktūros ir jungčių svorių keitimo uždavinys, siekiant, kad tinklas galėtų atlikti jam skirtą užduotį. Kiekviename neuroninio tinklo mokymo žingsnyje keičiamos svorių reikšmės atsižvelgiant į įėjimo ir išėjimo reikšmes, gautas ankstesniame mokymo žingsnyje. Procesas kartojamas, kol pasiekiamas norimas rezultatas.

Skirtingos tinklų architektūros reikalauja skirtingų jų mokymo algoritmų. Yra trys pagrindinės neuronų mokymo paradigmos:

1) mokymo su mokytoju algoritmai (*supervised learning*);
2) mokymo be mokytojo algoritmai (*unsupervised learning*);
3) hibridinis mokymas (*hybrid learning*).

Kalbant apie *mokymo su mokytoju algoritmus*, yra vartojama sąvoka *norimos išėjimo reikšmės* (*target, desired output*). Tai iš anksto žinomos reikšmės, pavyzdžiui, klasių numeriai, prognozuojamos reikšmės ir pan.

Mokymo su mokytoju atveju tinklo išėjimų reikšmės, skaičiuojamos kiekvienam įėjimo vektoriui $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $i \in \{1, \ldots, m\}$, yra tiesiogiai susijusios su norimomis tų išėjimų reikšmėmis. Tinklas koreguojamas keičiant svorių vektorių reikšmes ir siekiant gauti kiek galima mažesnę paklaidą, t. y. ieškoma tokių svorių, kad skirtumas tarp norimų išėjimo reikšmių ir reikšmių, gautų išmokius neuroninį tinklą, būtų kiek galima mažesnis.

Prie mokymo su mokytoju algoritmų priskiriamas perceptrono mokymas, nagrinėjamas 4.1.4 skyrelyje, o taip pat „klaidos skleidimo atgal“ algoritmas, pateikiamas 4.1.5 skyrelyje.

Kartais norimos gauti tinklo išėjimo reikšmės nėra žinomos. Tada naudojami *mokymo be mokytojo algoritmai*. Šio tipo metoduose tinklas mokomas ieškoti koreliacijų ar panašumų tarp mokymo aibės įėjimų. Čia nėra grįžtamojo ryšio, pasakančio, kuris atsakymas bus arba yra teisingas. Mokymo be mokytojo algoritmuose nėra mokytojo signalo. Turima tik mokymo aibė X, kuri sudaryta iš vektorių X_i, $i = 1, \ldots, m$, priklausančių erdvei R^n, t. y. $X = \{X_1, X_2, ..., X_m\}$. Metodų tikslas yra suskirstyti mokymo duomenis (vektorius) į tam tikras klases arba rasti juose kokius nors reguliarumus ar ypatumus. Gali būti sprendžiamas ir toks uždavinys: vektorius X_i, $i = 1, \ldots, m$, reikia atvaizduoti į mažesnio matmenų skaičiaus erdvės vektorių rinkinį taip,

kad X_i savybės būtų išlaikytos ir naujos aibės vektoriams [68] [70]. Mokymo be mokytojo sėkmę garantuoja nuo mokytojo nepriklausomas kriterijus, kuris mokymo metu optimizuojamas parenkant tinkamas tinklo jungčių svorių reikšmes.

Galimos trys mokymo be mokytojo strategijos:

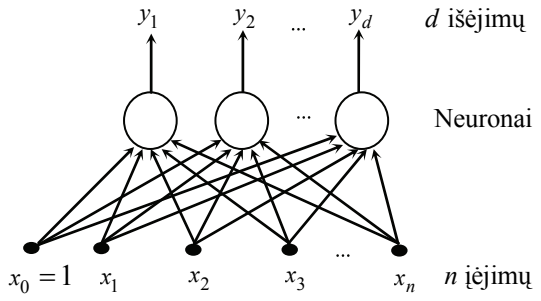
1) Hebbo mokymas (*Hebbian learning*);
2) varžytinių mokymas (*competitive learning*);
3) saviorganizuojantis mokymas (*self-organizing learning*).

Šios strategijos yra realizuojamos skirtingos struktūros neuroniniais tinklais ir todėl leidžia spręsti skirtingus duomenų analizės uždavinius.

Hibridinis mokymas apima mokymo su mokytoju ir be mokytojo algoritmus: dalis tinklo svorių nustatomi pagal mokymą su mokytoju, kita dalis gaunama iš mokymo be mokytojo.

4.1.4. Perceptronas, jo mokymas

Paprasčiausios architektūros yra vadinamieji *tiesioginio sklidimo* (*feedforward*) neuroniniai tinklai, kuriuose galimos tik vienkryptės jungtys į priekį (*unidirectional forward*) iš įėjimų į išėjimus. Paprasčiausias tokio tipo neuroninis tinklas yra *perceptronas*, sudarytas iš vieno sluoksnio *d* neuronų, sujungtų su *n* įėjimų (žr. 4.4 pav.) [131].



4.4 pav. Perceptronas

Neuronų skaičius *d* yra lygus išėjimų skaičiui. Kai kurie autoriai jau vieną neuroną vadina perceptronu [123], tada $d = 1$. Kiekvienas perceptrono išėjimas y_j, $j = 1,\ldots,d$, yra įėjimų $x_1, x_2, \ldots, x_n$ funkcija, kuri skaičiuojama pagal šią formulę:

$$y_j = f(a_j) = f\left(\sum_{k=0}^{n} w_{jk} x_k\right), \quad j = 1,\ldots,d\,, \tag{4.3}$$

čia w_{jk} yra jungties iš *k*-ojo įėjimo į *j*-ąjį neuroną svoris, x_0 – papildomas įėjimas, įprastai $x_0 = 1$.

Tegul turime *m* mokymo aibės vektorių $X_i = (x_{i1}, x_{i2},\ldots, x_{in})$, $i = 1,\ldots,m$. Vektorius X_i dar vadinsime įėjimo vektoriais, nes jo komponenčių reikšmės $x_{i1}, x_{i2},\ldots, x_{in}$ bus pateikiamos į neuroninį tinklą kaip įėjimai.

Įėjimo vektoriai $X_i = (x_{i1}, x_{i2},\ldots, x_{in})$, $i = 1,\ldots,m$, yra susieti su norimų reikšmių vektoriais $T_i = (t_{i1}, t_{i2},\ldots, t_{id})$, čia t_{ij} – norimo *j*-ojo išėjimo reakcija į X_i. Norimomis reikšmėmis gali būti klasių numeriai, prognozuojamos reikšmės ir kt.

Perceptrono mokymo procese svoriai w_{jk} keičiami taip, kad tinklo išėjimo reikšmių vektorius $Y_i = (y_{i1}, y_{i2},\ldots, y_{id})$, gautas į įėjimą pateikus vektorių X_i, būtų kiek galima artimesnis norimų reikšmių vektoriui T_i, t. y. perceptrono veikimo paklaida būtų kiek galima mažesnė.

Paklaidos matas $E(W)$ apibrėžiamas kaip svorių matricos $W = \{w_{jk},\ j = 1,\ldots,d,\ k = 0,\ldots,n\}$ funkcija. Jeigu paklaidos funkcija $E(W)$ yra diferencijuojama pagal svorius w_{jk}, jos minimumą galima rasti gradientiniais optimizavimo metodais. Geriausiai žinomas yra gradientinio nusileidimo algoritmas. Dažniausiai naudojama paklaidos funkcija yra kvadratinių paklaidų suma

$$E(W) = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{d}(y_{ij} - t_{ij})^2 = \sum_{i=1}^{m} E_i(W)\,,$$

$$E_i(W) = \frac{1}{2}\sum_{j=1}^{d}(y_{ij} - t_{ij})^2\,. \tag{4.4}$$

Iš pradžių generuojamos atsitiktinės svorių w_{jk} reikšmės. Tada gradientinio nusileidimo algoritmu judama antigradiento kryptimi, svorių reikšmes keičiant pagal iteracinę formulę

$$w_{jk}(t+1) = w_{jk}(t) + \Delta w_{jk}(t)\,, \tag{4.5}$$

čia

$$\Delta w_{jk}(t) = -\eta \frac{\partial E(t)}{\partial w_{jk}} = -\eta \sum_{i=1}^{m} \frac{\partial E_i(t)}{\partial w_{jk}} = \sum_{i=1}^{m} \Delta w_{jk}^{i}(t),$$

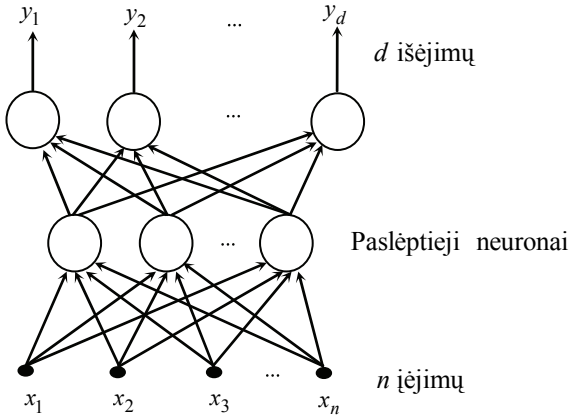
$$\Delta w_{jk}^{i}(t) = -\eta \frac{\partial E_i(t)}{\partial w_{jk}}, \tag{4.6}$$

η yra teigiamas daugiklis, kuris vadinamas *mokymo greičiu* (*learning rate*) ir kuriuo reguliuojamas gradientinio optimizavimo žingsnio ilgis, t – iteracijos numeris.

Galimos dvi svorių keitimo strategijos. Vienu atveju svoriai w_{jk} pakeičiami pagal (4.5) formulę pateikus į tinklą visus mokymo aibės vektorius. Kitoje strategijoje svoriai w_{jk} pakeičiami pagal (4.5) formulę po kiekvieno mokymo aibės vektoriaus pateikimo į tinklą.

4.1.5. Daugiasluoksniai tiesioginio sklidimo neuroniniai tinklai

Turintys daugiau nei vieną neuronų sluoksnį tinklai, kuriuose galimi tik ryšiai į priekį iš įėjimų į išėjimus, yra vadinami *daugiasluoksniais perceptronais* (*multilayer perceptrons*), arba *daugiasluoksniais tiesioginio sklidimo neuroniniais tinklais* (*multilayer feedforward neural networks*). Kiekvienas toks tinklas sudarytas iš įėjimų aibės, išėjimų neuronų sluoksnio ir paslėptųjų neuronų sluoksnių tarp įėjimų ir išėjimų (žr. 4.5 pav.).



4.5 pav. Tiesioginio sklidimo neuroninis tinklas

Kiekvienas neuronų sluoksnis turi po papildomą įėjimą ir jo jungtis su to sluoksnio neuronais, tačiau paprastumo dėlei jie 4.5 paveiksle nėra pavaizduoti.

Tegul turime daugiasluoksnį neuroninį tinklą, kuriame yra L sluoksnių, pažymėtų $l = 0,1,\ldots,L$, čia sluoksnis $l = 0$ žymi įėjimus, o $l = L$ – paskutinį (išėjimų) sluoksnį. Kiekviename sluoksnyje l yra n_l neuronų. Pirmojo sluoksnio *j*-ojo neurono išėjimo reikšmė y_j yra apskaičiuojama pagal (4.3) formulę. Įėjimai į neuronus *l*-ajame sluoksnyje yra neuronų išėjimai $(l-1)$-ajame sluoksnyje. Todėl kiekvieno *j*-ojo neurono išėjimo reikšmė y_j *l*-ajame sluoksnyje yra apskaičiuojama taip:

$$y_j = f(a_j) = f\left(\sum_{k=0}^{n_{l-1}} w_{jk} y_k\right), \quad j = 1,\ldots,n_l, \tag{4.7}$$

čia $f()$ yra neuronų aktyvacijos funkcija, w_{jk} – svoriai jungčių, kurios jungia *k*-ąjį neuroną $(l-1)$-ajame sluoksnyje su *j*-uoju neuronu l-ajame sluoksnyje, n_{l-1} – neuronų skaičius $(l-1)$-ajame sluoksnyje, $y_0 = 1$. Kairiojoje (4.7) lygybės pusėje y_j yra *l*-ojo sluoksnio *j*-ojo neurono išėjimo reikšmė, o dešiniojoje – y_k yra $(l-1)$-ojo sluoksnio *k*-ojo neurono išėjimo reikšmė. Indeksai l ir $(l-1)$ į (4.7) formulę neįtraukiami siekiant, kad būtų paprasčiau aiškinti tinklo mokymo algoritmą.

Pastebėsime, kad skirtingi neuronų sluoksniai arba net skirtingi neuronai gali turėti individualias aktyvacijos funkcijas. Tuo atveju pasikeistų ir (4.3) bei (4.7) formulės.

„Klaidos skleidimo atgal“ mokymo algoritmas

Taikant vienasluoksnio perceptrono mokymo idėją daugiasluoksniam neuroniniam tinklui, būtina žinoti paslėptųjų neuronų išėjimų reikšmes. Jei paklaidos ir aktyvacijos funkcijos yra diferencijuojamos, ieškant minimalios paklaidos gali būti naudojama gradientinio nusileidimo strategija. Algoritmas, kuris realizuoja gradientinio nusileidimo mokymo strategiją daugiasluoksniam tiesioginio sklidimo neuroniniam tinklui, vadinamas „*klaidos skleidimo atgal*“ *algoritmu* (*back-propagation learning algorithm*).

Algoritmą sudaro du žingsniai:

1) įėjimų reikšmių „skleidimas pirmyn“ iš įėjimų į išėjimų sluoksnį;
2) paklaidos „skleidimas atgal“ iš išėjimų į įėjimų sluoksnį.

Tiek perceptrono mokyme, tiek „klaidos skleidimo atgal“ algoritme naudojama mokymo su mokytoju strategija. Naudosimės (4.4) daline kvadratinių sumų paklaida $E_i(W)$, svorius keisime pagal (4.6) formulę. Pirmame algoritmo žingsnyje įėjimų vektoriui X_i apskaičiuojamas išėjimų vektorius Y_i. Įvertinama paklaidos funkcija $E_i(W)$ išėjimų sluoksnyje L. Tuo baigiama „skleidimo pirmyn“ fazė. Jei paklaidos funkcija $E_i(W)$ nelygi nuliui, reikia keisti jungčių svorius. Panašiai kaip ir vienasluoksniame perceptrone, visi svoriai w_{jk} jungčių, kurios jungia k-ąjį neuroną $(l-1)$-ajame sluoksnyje su j-uoju neuronu l-ajame sluoksnyje, keičiami naudojantis formule

$$\Delta w_{jk}^{i} = -\eta \frac{\partial E_i}{\partial w_{jk}}. \tag{4.8}$$

Dalinės išvestinės išreiškiamos taip:

$$\frac{\partial E_i}{\partial w_{jk}} = \frac{\partial E_i}{\partial a_{ij}} \cdot \frac{\partial a_{ij}}{\partial w_{jk}}. \tag{4.9}$$

Iš (4.7) formulės gauname

$$\frac{\partial a_{ij}}{\partial w_{jk}} = y_{ik}. \tag{4.10}$$

Tegul

$$\delta_{ij} = \frac{\partial E_i}{\partial a_{ij}}.$$

Įstatę šią ir (4.10) išraiškas į (4.9) ir (4.8) formules, gauname:

$$\frac{\partial E_i}{\partial w_{jk}} = \delta_{ij} y_{ik},$$

$$\Delta w_{jk}^{i} = -\eta \delta_{ij} y_{ik}, \tag{4.11}$$

čia j-asis neuronas priklauso l-ajam sluoksniui, k-asis neuronas priklauso $(l-1)$-ajam sluoksniui.

Išėjimų sluoksnyje

$$\delta_{ij} = \frac{\partial E_i}{\partial a_{ij}} = f'(a_{ij})(y_{ij} - t_{ij}), \tag{4.12}$$

čia j-asis neuronas priklauso išėjimų sluoksniui L.

Rasime $\frac{\partial E_i}{\partial a_{ij}}$ paslėptiesiems neuronams, t. y. kai *j*-asis neuronas priklauso *l*-ajam sluoksniui ir $l < L$. Naudojantis dalinėmis išvestinėmis, bendruoju atveju galima parašyti

$$\delta_{ij} = \frac{\partial E_i}{\partial a_{ij}} = \sum_{s=1}^{n_{l+1}} \frac{\partial E_i}{\partial a_{is}} \cdot \frac{\partial a_{is}}{\partial a_{ij}},$$

čia n_{l+1} žymi neuronų $(l+1)$-ajame sluoksnyje skaičių. Išraiška $\frac{\partial E_i}{\partial a_{is}}$ yra lygi dydžiui δ_{is}, apibrėžtam *s*-ajam neuronui $(l+1)$-ajame sluoksnyje. Atsižvelgdami į (4.7) formulę, gauname

$$\frac{\partial a_{is}}{\partial a_{ij}} = f'(a_{ij}) w_{is}.$$

Tada paslėptųjų *j*-ųjų neuronų

$$\delta_{ij} = f'(a_{ij}) \sum_{s=1}^{n_{l+1}} w_{is} \delta_{is}, \tag{4.13}$$

čia *j*-asis neuronas priklauso sluoksniui $l < L$, *s*-asis neuronas priklauso sluoksniui $l+1$.

Iš pradžių reikia apskaičiuoti δ_{ij} reikšmes išėjimų sluoksnyje *L* pagal (4.12) formulę. Tada palaipsniui skaičiuoti δ_{ij} reikšmes paslėptiems neuronams tarpiniuose sluoksniuose $l < L$ naudojantis $(l+1)$-ųjų sluoksnių δ_{ij} reikšmėmis, gautomis pagal (4.13) formulę.

Kai visi svoriai pakeičiami pagal (4.11) formulę, į tinklą pateikiamas sekantis mokymo vektorius X_i ir procesas kartojamas iš naujo. Algoritmo sustojimo kriterijus yra arba iš anksto nustatyta paklaidos funkcijos slenksčio reikšmė, arba atitinkamas atliktų iteracijų (mokymo žingsnių) skaičius.

Norint pagreitinti mokymo procesą, yra koreguojama taisyklė, apibrėžta (4.11) formule. Gaunama tokia svorių pokyčio formulė:

$$\Delta w^i_{jk}(t) = -\eta \delta_{ij}(t) y_{ik}(t) + \alpha \Delta w^i_{jk}(t-1),$$

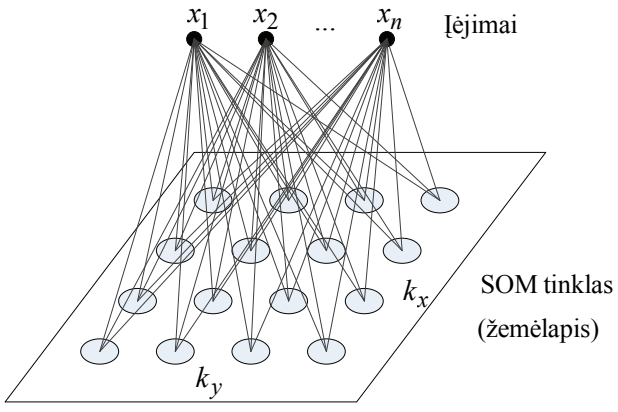
čia *t* yra iteracijos numeris, α – teigiama konstanta ($0 < \alpha \le 1$), vadinamoji *momento konstanta* (*momentum constant*).

4.2. Saviorganizuojantys neuroniniai tinklai

Saviorganizuojančius neuroninius tinklus (*žemėlapius*, *self-organizing maps*, SOM) T. Kohonenas pradėjo tyrinėti apie 1982 metus [87]. Jie dar vadinami Kohoneno neuroniniais tinklais, arba Kohoneno saviorganizuojančiais žemėlapiais. Nors pagrindiniai SOM tyrimų darbai vyksta Helsinkio technologijos universiteto Kompiuterių ir informacijos mokslų laboratorijos Neuroninių tinklų tyrimų centre (http://www.cis.hut.fi), tačiau dabar SOM tinklai yra nagrinėjami daugelio pasaulio mokslininkų ir plačiai taikomi įvairiose srityse.

Šio tipo neuroninių tinklų pavadinimas kilo iš to, kad saviorganizuojantis žemėlapis, naudodamas mokymo (įėjimo) aibę, pats save sukuria (save organizuoja). Pagrindinis SOM tinklo tikslas – išlaikyti duomenų topologiją. Taškai, esantys arti įėjimo vektorių erdvėje, yra atvaizduojami arti vieni kitų ir SOM tinkle. SOM tinklai gali būti naudojami siekiant vizualiai pateikti duomenų klasterius ir ieškant daugiamačių duomenų projekcijų į mažesnio skaičiaus matmenų erdvę, paprastai į plokštumą.

Saviorganizuojantis neuroninis tinklas yra neuronų, paprastai išdėstytų dvimačio tinklelio, dar vadinamo žemėlapiu arba lentele, mazguose, masyvas $M = \{M_{ij}, i = 1,\ldots,k_x, j = 1,\ldots,k_y\}$. Dažniausiai yra analizuojami dvimačiai SOM tinklai, nors galimi ir didesnio matmenų skaičiaus tinklai. Dvimačio neuroninio tinklo schema pateikta 4.6 paveiksle.



4.6 pav. Dvimačio SOM tinklo schema

Būtina suprasti, kad po kiekvienu SOM tinklo neuronu (paveiksle pažymėtu apskritimu) „slepiasi“ vektorius (*codebook vector*), kurio matmenų skaičius sutampa su analizuojamos aibės vektorių matmenų skaičiumi.

Tegul turime *n*-mačių įėjimo vektorių aibę $X = \{X_1, X_2, \ldots, X_m\}$ ($X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $i = 1, \ldots, m$), kuri bus naudojama SOM tinklui mokyti. Kiekvienas žemėlapio neuronas sujungtas su kiekviena įėjimo vektoriaus komponente (žr. 4.6 pav.). Priešingai nei anksčiau nagrinėti tiesioginio sklidimo neuroniniai tinklai, nulinio įėjimo SOM tinklas neturi.

Galima stačiakampė (*rectangular*, žr. 4.7a pav.) arba šešiakampė (*hexagonal*, žr. 4.7b pav.) tinklo struktūra. Čia detaliau analizuosime stačiakampę tinklo struktūrą, kai k_x yra skaičius lentelės eilučių, k_y – stulpelių, o iš viso neuronų yra $k_x \times k_y$.

a) b)

4.7 pav. SOM tinklo struktūra: a) stačiakampė, b) šešiakampė

4.2.1. SOM tinklo mokymas

Neuroninį tinklą mokysime *n*-mačiais vektoriais $X_1, X_2, \ldots, X_m$, kuriuos vadinsime duomenų, mokymo arba įėjimo vektoriais. Kiekvienas šios aibės vektorius mokymo metu yra susiejamas su vienu tinklo neuronu, kuris taip pat yra *n*-matis vektorius. Vektorių, nusakantį (apibūdinantį) *i*-osios eilutės *j*-ajame stulpelyje esantį neuroną, pažymėkime $M_{ij} = (m_1^{ij}, m_2^{ij}, \ldots, m_n^{ij}) \in R^n$.

SOM tinklas mokomas mokymo be mokytojo būdu. Mokymo pradžioje neuronų (vektorių) M_{ij} komponenčių $m_1^{ij}, m_2^{ij}, \ldots, m_n^{ij}$ pradinės reikšmės dažniausiai nustatomos atsitiktinai. Neuroniniam tinklui daug kartų pateikiama skirtingų objektų, nusakomų *n*-mačiais vektoriais $X_1, X_2, \ldots, X_m$:

- Kiekviename mokymo žingsnyje (iteracijoje) vienas mokymo aibės vektorius $X_k \in \{X_1, X_2, \ldots, X_m\}$ pateikiamas į tinklą.

- Vektorius X_k palyginamas su visais neuronais M_{ij}: dažniausiai skaičiuojamas Euklido atstumas ($\| X_k - M_{ij} \|$) tarp šio vektoriaus X_k ir kiekvieno neurono M_{ij}.
- Randama, iki kurio neurono $M_c \in \{M_{ij}, i = 1,\ldots,k_x, j = 1,\ldots,k_y\}$ atstumas yra mažiausias; rastas neuronas M_c vadinamas *neuronu* (*vektoriumi*) *nugalėtoju* (*neuron* (*vector*) *winner*). Pažymėkime i_c eilutę, o j_c stulpelį, kurie nusako neurono M_c vietą tinkle, čia *c* yra skaičių i_c ir j_c pora, t. y.

$$c = \arg\min_{i,j} \left\{ \| X_k - M_{ij} \| \right\}, \quad \| X_k - M_c \| = \min_{i,j} \left\{ \| X_k - M_{ij} \| \right\}.$$

- Visų tinklo neuronų komponentės keičiamos naudojantis iteracine formule

$$M_{ij}(t+1) = M_{ij}(t) + h_{ij}^c(t)\left(X_k - M_{ij}(t) \right).$$

Šioje formulėje *t* yra iteracijos numeris, $h_{ij}^c(t)$ – *kaimynystės funkcija*:

$$h_{ij}^c(t) = h_{ij}^c\left(\| \hat{R}_c - \hat{R}_{ij} \|, t \right),$$

čia $\hat{R}_c$ ir $\hat{R}_{ij}$ yra dvimačiai vektoriai, sudaryti iš M_c ir M_{ij} indeksų (eilučių ir stulpelių numerių), nusakančių vektoriaus X_k neurono nugalėtojo M_c ir perskaičiuojamo neurono M_{ij} vietą SOM tinkle. Procesui konverguoti būtina, kad $h_{ij}^c(t) \to 0$, kai $t \to \infty$. Dydis $\| \hat{R}_c - \hat{R}_{ij} \|$ yra Euklido atstumas tarp vektorių $\hat{R}_c$ ir $\hat{R}_{ij}$. Jam didėjant, funkcijos $h_{ij}^c(t)$ reikšmė artėja prie nulio ($h_{ij}^c(t) \to 0$).

SOM mokyme vienos iteracijos metu į tinklą pateikiamas vienas vektorius. Norint tinklą geriau išmokyti, tikslinga kiekvieną vektorių į tinklą pateikti kelis kartus. Galimi trys būdai:

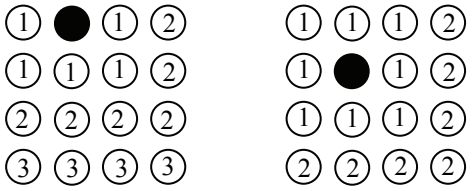
- įėjimo aibės vektoriai pateikiami iš eilės po vieną ciklišk ai, t. y. pateikus visus vektorius, vėl pirmasis pateikiamas į tinklą ir t. t.;
- vektoriai pateikiami atsitiktine tvarka, t. y. vektoriai sumaišomi ir tada vienas po kito pateikiami į tinklą; kai visi jau pateikti, permaišomi ir vėl pateikiami į tinklą ir t. t;
- į tinklą pateikiamas atsitiktinai paimtas vienas įėjimo aibės vektorius, vėliau vėl atsitiktinai imamas kitas ir t. t.

Pirmais dviem atvejais visi vektoriai pateikiami vienodą skaičių kartų, trečiuoju – nebūtinai. Antro būdo privalumas prieš pirmąjį yra tai, kad išeliminuojama galimybė tinklui „prisiminti“ įėjimo vektorių pateikimo į tinklą tvarką.

Kartais yra vartojamas terminas *epocha*. *Viena epocha* – tai mokymo proceso dalis, kurios metu visi mokymo aibės vektoriai nuo X_1 iki X_m (m – vektorių skaičius) po vieną kartą pateikiami į tinklą nuosekliai arba atsitiktine tvarka.

Po SOM tinklo mokymo į tinklą pateikiami mokymo aibės arba nauji, dar tinklui „nematyti“, duomenų vektoriai. Randamas kiekvieno vektoriaus neuronas nugalėtojas ir jis pažymimas SOM žemėlapyje neurono nugalėtojo vietoje. Tokiu būdu vektoriai išsidėsto tarp žemėlapio (lentelės) elementų.

Apibrėžkime dar kelias su SOM tinklo struktūra susijusias sąvokas: *neurono kaimynas*, *kaimynystės eilė*. Greta vektoriaus nugalėtojo M_c esantys neuronai vadinami *pirmosios eilės kaimynais* (kaimynystės eilė $\eta_{ij}^c = 1$). Greta pirmosios eilės kaimynų esantys neuronai, išskyrus jau paminėtus, vadinami *antrosios eilės kaimynais* (kaimynystės eilė $\eta_{ij}^c = 2$) ir t. t. Kaimynų eilės pažymėtų neuronų atžvilgiu parodytos 4.8 paveiksle. Kaimynystės eilė η_{ij}^c gali būti integruojama į kaimynystės funkciją vietoj atstumų $\| \hat{R}_c - \hat{R}_{ij} \|$.



4.8 pav. Kaimynų eilės pažymėtų neuronų atžvilgiu

Yra įvairių SOM tinklo mokymo realizacijų, kurios viena nuo kitos skiriasi kaimynystės funkcijos $h_{ij}^c(t)$ išraiška. Tai yra euristinės funkcijos, todėl griežtų matematinių konvergavimo įrodymų nėra ir skirtingų mokymo taisyklių rezultatai gali būti šiek tiek kitokie žemėlapiai. Stabilios analizuojamų duomenų grupės įprastai išlieka visuose žemėlapiuose, tačiau gali būti duomenų, kurie priskiriami vis prie kitų grupių arba visai jų nesudaro. Tačiau tai yra savotiškas metodo privalumas, nes pagrindinis vizualizavimo tikslas yra

padėti suvokti analizuojamus duomenis, atskleisti jų struktūrą, kelti hipotezes dėl analizuojamų duomenų aibės. Keli gauti vaizdai tai padaryti padeda daug efektyviau.

Viena iš galimų kaimynystės funkcijos h_{ij}^c išraiškų yra tokia [37]:

$$h_{ij}^c = \frac{\alpha}{\alpha \eta_{ij}^c + 1},$$

čia

$$\alpha = \max\left(\frac{e + 1 - \hat{e}}{e}; 0{,}01 \right),$$

e – prieš tinklo mokymą nustatytas viso mokymo epochų skaičius, $\hat{e}$ – vykdomos epochos numeris, η_{ij}^c – kaimynystės tarp M_c ir M_{ij} eilė.

Neurono nugalėtojo M_c kaimynystės funkcijos h_{ij}^c reikšmė yra maksimali. Ji mažėja augant epochų eilės numeriui $\hat{e}$ ir didėjant kaimynystės eilei η_{ij}^c nugalėtojo atžvilgiu.

Kiekvienos epochos metu perskaičiuojami tie neuronai M_{ij}, kuriems galioja nelygybė

$$\eta_{ij}^c \le \max\left[\alpha \max(k_x, k_y); 1 \right].$$

Mokymo pradžioje perskaičiuojami ir tolimesni kaimynai, vėliau tik artimesni.

4.2.2. SOM tinklo mokymo kokybės nustatymas

Baigus SOM tinklo mokymus, būtina nustatyti jo kokybę. Dažniausiai vertinamos dvi paklaidos: *kvantavimo* (*quantization error*) ir topografinė (*topographic error*).

Kvantavimo paklaida parodo, kaip tiksliai jau išmokyto tinklo neuronai prisiderina prie mokymo aibės vektorių. Jei visi vektorių X_k neuronai nugalėtojai $M_{c(k)}$ būtų lygiai tokie pat kaip ir patys vektoriai X_k, tai kvantavimo paklaida būtų lygi 0.

Kvantavimo paklaida $E_{SOM(q)}$ – tai vidutinis atstumas tarp duomenų vektorių X_k ir jų vektorių nugalėtojų $M_{c(k)}$:

$$E_{SOM(q)} = \frac{1}{m} \sum_{k=1}^{m} \| X_k - M_{c(k)} \| . \tag{4.14}$$

Topografinė paklaida parodo, kaip gerai SOM tinklas išlaiko analizuojamų duomenų topografiją, t. y. tarpusavio išsidėstymą. Topografinė paklaida $E_{SOM(t)}$ skaičiuojama pagal šią formulę:

$$E_{SOM(t)} = \frac{1}{m} \sum_{k=1}^{m} u(X_k) . \tag{4.15}$$

Jeigu SOM žemėlapyje vektoriaus X_k neuronas nugalėtojas yra šalia neurono, iki kurio atstumas nuo X_k yra mažiausias, neskaičiuojant iki neurono nugalėtojo, tai (4.15) formulėje $u(X_k) = 0$, priešingu atveju $u(X_k) = 1$.

4.2.3. SOM tinklas daugiamačiams duomenims vizualizuoti

Saviorganizuojantis neuroninis tinklas yra tinkamas daugiamačių duomenų vizualizavimo įrankis, galintis daugiamačius duomenis ne tik atvaizduoti plokštumoje, bet prieš tai juos ir klasterizuoti. Tai parodyta [52] darbe.

Baigus SOM tinklo mokymus, analizuojami vektoriai (mokymo ar naujos aibės) pateikiami į tinklą. Kiekvienam vektoriui randamas neuronas nugalėtojas. Vektorių numeriai, klasių, kurioms jie priklauso, pavadinimai ar kita informacija apie vektorius užrašomi tuose žemėlapio (lentelės) langeliuose, kuriuos atitinka jų neuronai nugalėtojai. Taigi SOM tinkle daugiamačiai duomenys transformuojami į tam tikrą diskrečią struktūrą. Tokiu būdu vektoriai išsidėsto tarp žemėlapio elementų. Tai galima laikyti kaip *n*-mačių vektorių (taškų) išsidėstymą plokštumoje. Jų vietą plokštumoje nusako tinklo mazgai – eilučių ir stulpelių numeriai. Paprasčiausiu (stačiakampės topologijos) atveju gaunama lentelė, kurios langeliuose surašyti analizuojamų vektorių numeriai arba klasių pavadinimai.

Irisų duomenis atitinkančių vektorių $X_1, X_2, \ldots, X_{150}$ išsidėstymas SOM tinkle pateiktas 4.1 lentelėje. Čia skaičiai yra irisų klasių numeriai. Reikia nepamiršti, kad į SOM tinklą atitinkančios lentelės atskirą langelį gali patekti ne vienas, bet keli analizuojami vektoriai, tarp jų ir priklausantys skirtingoms klasėms. Matome, kad pirmos klasės (*Setosa*) irisai sudaro ryškiai išsiskiriančią grupę. Antros ir trečios klasių (*Versicolor* ir *Virginica*) irisai irgi su nedidelėmis išimtimis sudaro atskirus, tačiau sugludusius vienas šalia kito klasterius.

4.1 lentelė. Irisų duomenys, pateikti SOM tinkle [10x10]

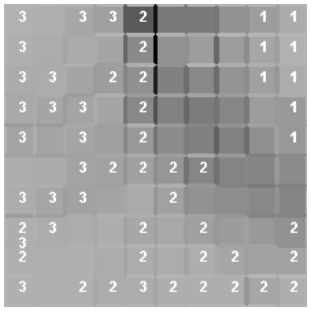
3		3	3	2				1	1
3				2				1	1
3	3		2	2				1	1
3	3	3		2					1
3		3		2					1
		3	2	2	2	2			
3	3	3			2				
2,3	3			2		2			2
2				2		2	2		2
3		2	2	3	2	2	2	2	2

Tokia lentelė nėra labai informatyvi, sunku pasakyti, kaip toli yra vektoriai, esantys gretimuose lentelės langeliuose. Todėl būtina ieškoti būdų, kaip pagerinti tokio vizualizavimo kokybę.

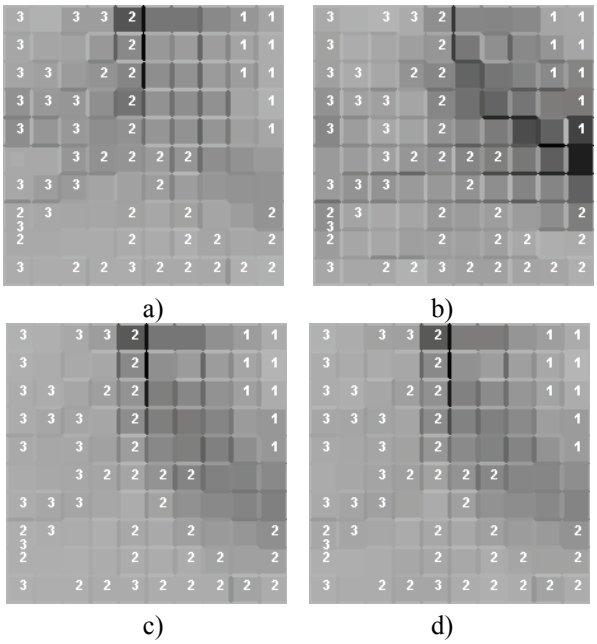
Unifikuota atstumų matrica (*U-matrica*, *unified distance matrix*) yra vienas iš populiaresnių SOM tinklo vizualizavimo metodų. U-matricą sudaro atstumai tarp kaimyninių SOM neuronų. Paprastumo dėlei pateikiamas vienmačio SOM tinklo pavyzdys. Turint [1x5] didumo (1 eilutės ir 5 stulpelių) tinklą $(M_1, M_2, \ldots, M_5)$ U-matrica yra vienos eilutės ir devynių stulpelių vektorius $(u_1, u_{12}, u_2, u_{23}, u_3, u_{34}, u_4, u_{45}, u_5)$. Čia $u_{ij} = \| M_i - M_j \|$ yra atstumas tarp dviejų kaimyninių neuronų M_i ir M_j, o u_i yra tam tikra apibrėžta reikšmė, pavyzdžiui, vidutinis atstumas tarp kaimyninių reikšmių:

$$u_i = \frac{u_{(i-1)i} + u_{i(i+1)}}{2}.$$

Radus U-matricą, jos reikšmes reikia tam tikru būdu pavaizduoti SOM tinkle. Darbuose [90], [141] pasiūlytas metodas, pagal kurį vidutiniai atstumai tarp kaimyninių neuronų yra pateikiami pilkos skalės atspalviais (vėliau imta naudoti ir kitų spalvų skales). Jei vidutiniai atstumai tarp kaimyninių neuronų yra maži, tuos neuronus atitinkantys tinklo langeliai spalvinami šviesia spalva; tamsi spalva reiškia didelius atstumus. Klasteriai yra nustatomi pagal šviesius atspalvius, o jų ribos – pagal tamsesnius [87], [88]. U-matrica parodyta 4.9 paveiksle. Matyti, kad pirmos klasės irisai atsiskiria nuo kitų dviejų, griežtos ribos tarp antros ir trečios klasių nėra. U-matrica gauta naudojantis *Nenet* sistema (http://koti.mbnet.fi/~phodju/nenet/Nenet/General.html).



4.9 pav. U-matrica, gauta analizuojant irisų duomenis

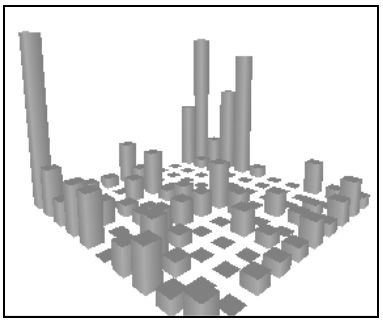


4.10 pav. Komponenčių plokštumos, gautos analizuojant tik irisų:
a) taurėlapių ilgius, b) taurėlapių pločius, c) vainiklapių ilgius, d) vainiklapių pločius

Kiekviena neurono komponentė atitinka skirtingas įėjimo vektoriaus komponentes. Taikant U-matricos vizualizavimo idėją, galima atvaizduoti atskiras komponentes. Tokiu būdu gaunamos vadinamosios *komponenčių plokštumos* (*component planes*), kurios pavaizduotos 4.10 paveiksle. Iš jo galima matyti, kokią įtaką klasterizavimui daro viena ar kita analizuojamų vektorių komponentė.

Dažniausiai yra naudojama dvimatė U-matrica, tačiau [134] darbe siūloma taikyti trimatį SOM tinklo vizualizavimą, kuris padeda geriau atskirti susidariusius klasterius.

SOM tinklui vizualizuoti gali būti naudojamos *histogramos*. Histograma parodo, kiek vektorių priklauso klasteriui, apibrėžtam vienu neuronu. Randami kiekvieno duomenų vektoriaus neuronai nugalėtojai, po to suskaičiuojama, kelių vektorių neuronai nugalėtojai yra tie patys. Galimi keli histogramų vizualizavimo būdai. Vienas jų pateiktas 4.11 paveiksle.



4.11 pav. Neuronų nugalėtojų histograma, gauta analizuojant irisų duomenis

4.2.4. SOM tinklų programiniai resursai

Sparčiai vystantis saviorganizuojančių neuroninių tinklų teorijai ir taikymams, vis daugiau sukuriama ir juos realizuojančių programinių sistemų. Kai kurios iš jų laisvai platinamos arba jų demonstracinės versijos yra visiems prieinamos internete. Sistemos skiriasi viena nuo kitos realizacijos, tinklo mokymo taisyklėmis ir vizualizavimo galimybėmis. Šiame skyrelyje nagrinėjama sistemų įvairovė, atskleidžiami jų privalumai ir trūkumai, pateikiama šių sistemų lyginamoji analizė.

Analizuojamos SOM tinklų sistemos

Analizuojamos šios sistemos:

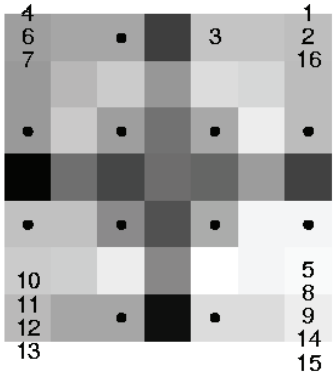
- sistema *SOM-PAK*, sukurta Helsinkio technologijos universiteto Kompiuterių ir informacijos mokslų laboratorijos Neuroninių tinklų tyrimų centre (http://www.cis.hut.fi/research/som_lvq_pak.shtml);
- sistema *SOM-Toolbox*, sukurta tame pačiame centre, kaip ir sistema *SOM-PAK* (http://www.cis.hut.fi/research/som_lvq_pak.shtml);

- sistema *Viscovery SOMine*; eksperimentai buvo atliekami naudojantis 3.0 versija (*Standard Edition Version 3.0*), kurios bandomoji versija internete buvo platinama nemokomai (http://www.viscovery.net);
- sistema *Nenet*, kurios demonstracine versija galima naudotis nemokamai (http://koti.mbnet.fi/~phodju/nenet/Nenet/General.html);
- *Kleiwego* sistema, sukurta Olandijos Groningeno universitete (http://odur.let.rug.nl/~kleiweg/kohonen/kohonen.html).

Gautų SOM tinklų analizė

Lyginant sistemų galimybes, analizuoti ekologiniai duomenys, nusakantys Suomijos pajūrio kopas ir jų vegetaciją [71] (detalus duomenų aprašas pateiktas 5.3.4 skyrelyje). Sistemose naudojamos įvairios U-matricos realizacijos [41]. Jų grafiniai rezultatai pateikti 4.12–4.16 paveiksluose. Eksperimentuose su visomis sistemomis, išskyrus *SOM-Toolbox*, naudotas stačiakampis [4x4] SOM tinklas. Sistemoje *SOM-Toolbox* leidžiama tik šešiakampė tinklo struktūra.

Sistemoje *SOM-PAK* analizuojamos aibės vektoriai išdėstomi [4x4] dydžio SOM tinkle (žr. 4.12 pav.). Tinklą atitinkančioje lentelėje yra dviejų tipų langeliai: vieni iš jų, vadinamieji tinklo mazgai (pažymėti taškais arba analizuojamų vektorių numeriais), atitinka neuronus, kiti – tarpinius langelius. Tarpinių langelių atspalvis nusako atstumus tarp gretimų tinklo neuronų: šviesesni tarpiniai langeliai tarp tinklo mazgų reiškia, kad mazgus atitinkantys neuronai yra artimesni nei tie, tarp kurių tarpiniai langeliai yra tamsesni. Tinklo mazgų (neuronų) spalva priklauso nuo gretimus tarpinius langelius atitinkančių U-matricos elementų reikšmių.

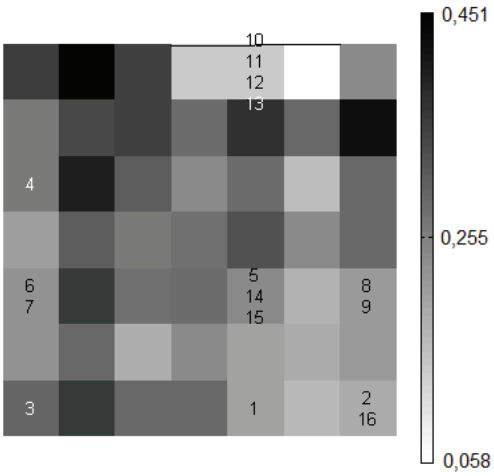


4.12 pav. Suomijos pajūrio kopas ir jų vegetaciją apibūdinančių parametrų išsidėstymas SOM [4x4] tinkle, gautame sistema *SOM-PAK*

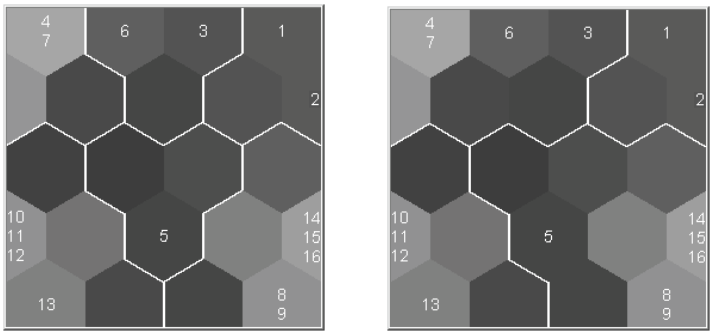
Sistemoje *SOM-Toolbox* buvo mokomas [4×4] tinklas. Tačiau, kaip matyti iš 4.13 paveikslo, gautas [7×7] tinklas. Čia, kaip ir sistemoje *SOM-PAK*, gaunamas tinklas su tarpiniais langeliais, tačiau jie niekaip neatskirti nuo tinklo mazgų (neuronų). Mazgų ir tarpinių langelių spalva parenkama kaip ir sistemoje *SOM-PAK*.

Sistemoje *Viscovery SOMine* leidžiama tik šešiakampė tinklo struktūra. Sistema automatiškai gali išskirti klasterius ir nubrėžti tarp jų linijas. Analizuojamuose duomenyse dauguma nagrinėjamų sistemų rado keturis klasterius, todėl su sistema *Viscovery SOMine* buvo atlikti du eksperimentai: sistema pati automatiškai išskyrė 6 klasterius (žr. 4.14a pav.); buvo reikalauta, kad sistema išskirtų tik 4 klasterius (žr. 4.14b pav.).

Nenet ir *Kleiwego* sistemose taip pat naudojamos U-matricos idėjos (žr. 4.15 ir 4.16 pav.), tačiau *Kleiwego* sistemoje skirtingais atspalviais spalvinamos tik neuronus skiriančios sienelės pagal atstumus tarp kaimyninius neuronus apibūdinančių vektorių: tamsiausia spalva atitinka didžiausią skirtumą, šviesiausia – didžiausią panašumą. Taip pat šioje sistemoje skaičiuojami mažiausi atstumai tarp neuronus nugalėtojus nusakančių vektorių ir pagal juos kuriamas minimalaus jungimo medis (*minimal spaning tree*), t. y. visi langeliai, kurie yra užpildyti analizuojamų vektorių numeriais, sujungiami linijomis įvertinant atstumus tarp neuronų nugalėtojų. Linijos yra brūkšninės, skirtingo žingsnio. Kuo linija vientisesnė, tuo atstumas tarp tų vektorių yra mažesnis.

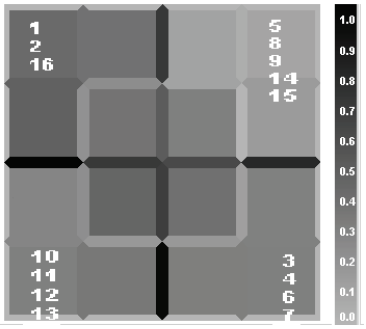


4.13 pav. Suomijos pajūrio kopas ir jų vegetaciją apibūdinančių parametrų išsidėstymas SOM [4x4] tinkle, gautame sistema *SOM-TOOLBOX*

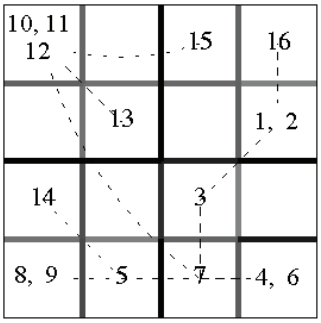


a) b)

4.14 pav. Suomijos pajūrio kopas ir jų vegetaciją apibūdinančių parametrų išsidėstymas SOM tinkluose, gautuose sistema *Viscovery SOMine*:
a) automatiškai išskiriami 6 klasteriai, b) priverstinai išskiriami 4 klasteriai



4.15 pav. Suomijos pajūrio kopas ir jų vegetaciją apibūdinančių parametrų išsidėstymas SOM [4x4] tinkle, gautame sistema *Nenet*



4.16 pav. Suomijos pajūrio kopas ir jų vegetaciją apibūdinančių parametrų išsidėstymas SOM [4x4] tinkle, gautame *Kleiwego* sistema

Kokia sistema naudotasi ir kokie Suomijos pajūrio kopas bei jų vegetaciją apibūdinančių parametrų klasteriai susidarė, parodyta 4.2 lentelėje.

Galima daryti bendras išvadas, kad:

- neabejotinai tvirtas klasteris yra $\{x_{10}, x_{11}, x_{12}, x_{13}\}$, kuris susidaro visuose SOM tinklų žemėlapiuose;
- gana tvirti klasteriai yra $\{x_5, x_8, x_9, x_{14}, x_{15}\}$, $\{x_3, x_4, x_6, x_7\}$ ir $\{x_1, x_2, x_{16}\}$.

4.2 lentelė. Suomijos pajūrio kopas ir jų vegetaciją apibūdinančių parametrų klasteriai

Sistema	**Susidarę klasteriai**
SOM-PAK	Keturi klasteriai: $\{x_1, x_2, x_3, x_{16}\}$, $\{x_4, x_6, x_7\}$, $\{x_5, x_8, x_9, x_{14}, x_{15}\}$, $\{x_{10}, x_{11}, x_{12}, x_{13}\}$. Parametras x_3 linkęs sudaryti atskirą klasterį.
SOM-Toolbox	Vienas klasteris yra $\{x_{10}, x_{11}, x_{12}, x_{13})$. Galima manyti, kad kitas didelis klasteris yra $\{x_1, x_2, x_5, x_8, x_9, x_{14}, x_{15}, x_{16}\}$, tačiau jis gali suskilti į du klasterius – $\{x_1, x_2, x_{16}\}$ ir $\{x_5, x_8, x_9, x_{14}, x_{15}\}$. Trečias didelis klasteris yra $\{x_3, x_4, x_6, x_7\}$, tačiau parametrai šiame klasteryje nėra taip stipriai susiję kaip kituose klasteriuose.
Viscovery SOMine	Keturi klasteriai: $\{x_1, x_2\}$, $\{x_3, x_4, x_6, x_7\}$, $\{x_{10}, x_{11}, x_{12}, x_{13}\}$, $\{x_5, x_8, x_9, x_{14}, x_{15}, x_{16}\}$. Arba šeši klasteriai: $\{x_1, x_2\}$, $\{x_3, x_6\}$, $\{x_4, x_7\}$, $\{x_{10}, x_{11}, x_{12}, x_{13}\}$, $\{x_5\}$, $\{x_8, x_9, x_{14}, x_{15}, x_{16}\}$.
Nenet	Keturi klasteriai: $\{x_3, x_4, x_6, x_7\}$, $\{x_5, x_8, x_9, x_{14}, x_{15}\}$, $\{x_{10}, x_{11}, x_{12}, x_{13}\}$, $\{x_1, x_2, x_{16}\}$.
Kleiwego sistema	Keturi klasteriai: $\{x_3, x_4, x_6, x_7\}$, $\{x_5, x_8, x_9, x_{14}\}$, $\{x_{10}, x_{11}, x_{12}, x_{13}, x_{15}\}$, $\{x_1, x_2, x_{16}\}$.

Tyrimai parodė, kad šiuo metu yra nemažai saviorganizuojančius neuroninius tinklus realizuojančių sistemų, turinčių savo privalumų ir trūkumų (jie nurodyti 4.3 lentelėje). Iš 4.12–4.16 paveikslų matyti, kad visomis sistemomis gauti rezultatai gana panašūs. Sistemomis *SOM-PAK* ir *SOM-Toolbox* gautų žemėlapių vizualizavimo būdas yra labai panašus. *SOM-Toolbox*

pranašumas tas, kad prie žemėlapio pateikiama atspalvių paletė su joje nurodomais skaitiniais dydžiais. Profesionalams tinkamesnės yra sistemos *SOM-Toolbox*, *Viscovery SOMine*, mažiau įgudusiems vartotojams – sistema *Nenet*.

4.3 lentelė. Analizuojamų sistemų privalumai ir trūkumai

Sistema	**Privalumai**	**Trūkumai**
SOM-PAK	Galima keisti nemažai parametrų. Yra interaktyvi programa, kuri inicializuoja pradinį žemėlapį ir jį moko.	Veikia tik *MS-DOS* ir *Unix* terpėse. Gaunamus rezultatus gana sunku interpretuoti. Parametrus būtina nurodyti komandinėje eilutėje.
SOM-Toolbox	Daug įvairių funkcijų. Įvairūs rezultatų vizualizavimo būdai.	Būtina *Matlab* sistema ir nors minimalios žinios apie ją.
Viscovery SOMine	Turi savo grafinę *Windows* sąsają.	Bandomoji versija yra ribotų galimybių.
Nenet	Turi savo grafinę *Windows* sąsają. Patogi vartotojui, net pradedančiajam.	Bandomoji versija yra ribotų galimybių.
Kleiweg sistema	Gaunamus rezultatus gana lengva interpretuoti. Nereikia įvedinėti daug parametrų.	Galima mokyti tik stačiakampės topologijos tinklą (žemėlapį). Mažai keičiamų parametrų. Veikia tik *MS-DOS* ir *Unix* terpėse. Parametrus būtina nurodyti komandinėje eilutėje.

4.3. SOM tinklo ir Sammono algoritmo jungimo būdai

Kaip jau minėta, kiekvieną SOM žemėlapio elementą (neuroną) atitinka *n*-matis vektorius. Stačiakampis tinklelis (žemėlapis) yra sudarytas iš $k_x \times k_y$ elementų – *n*-mačių vektorių (k_x eilučių ir k_y stulpelių). Neuroninis tinklas mokomas, jam daug kartų pateikiant *m* skirtingų objektų $X_1, X_2, \ldots, X_m$, nusakomų *n*-mačiais vektoriais. Mokant tinklą apskaičiuojami žemėlapio vektoriai ir tuos vektorius atitinkančių objektų numeriai, t. y. objektai pasiskirsto tarp žemėlapio elementų. Keli žemėlapio elementai gali likti nesusiję su jokiais analizuojamos aibės $X = \{X_1, X_2, \ldots, X_m\}$ vektoriais, tačiau

kai kurie gali būti susiję su keliais įėjimo vektoriais. Stačiakampės tinklo topologijos atveju galima nubraižyti lentelę, kurios langeliai atitinka neuronus. Tai galima laikyti kaip *n*-mačių taškų išsidėstymą plokštumoje. Jų vietą plokštumoje nusako tinklelio mazgai, t. y. eilučių ir stulpelių numeriai. Langeliai, atitinkantys neuronus nugalėtojus, užpildomi vektorių $X_1, X_2, \ldots, X_m$ numeriais, keli langeliai lieka tušti. Tačiau iš lentelės neaišku, kaip arti kaimyniniuose langeliuose esantys vektoriai yra *n*-matėje erdvėje. Jau 4.2.3 ir 4.2.4 skyreliuose nagrinėjome tokios SOM lentelės papildomo vizualizavimo būdus, tačiau jie turi trūkumą – ne visada aiškiai atskleidžiama analizuojamų duomenų struktūra, pavyzdžiui, klasteriai, jų artumas.

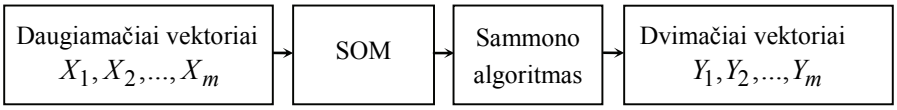
Kartais gautus rezultatus gana sudėtinga interpretuoti. Reikia ieškoti būdų, palengvinančių juos suvokti. Žemėlapio elementus apibūdina *n*-mačiai vektoriai $M_{ij} = (m_1^{ij}, m_2^{ij}, \ldots, m_n^{ij}) \in R^n$, todėl kyla idėja juos analizuoti vienu iš daugiamačių duomenų projekcijos metodu. Sammono algoritmas, kaip vienas iš daugiamačių skalių tipo metodų, puikiai tinka šiam tikslui pasiekti. Galimi keli SOM tinklo ir Sammono algoritmo jungimo būdai:

- nuoseklusis jungimas;
- integruotasis jungimas.

Nuosekliojo SOM tinklo ir Sammono algoritmų junginio pradžioje SOM tinklas mokomas *n*-mačiais vektoriais. Tada gauti *n*-mačiai vektoriai nugalėtojai Sammono algoritmu atvaizduojami plokštumoje [37], [82], [89]. Integruotajame junginyje daugiamačiai vektoriai yra atvaizduojami plokštumoje Sammono algoritmu atsižvelgiant į SOM tinklo mokymosi eigą [44], [96]. Junginiuose vietoj Sammono algoritmo gali būti naudojamas ir kitas daugiamačių skalių (DS) grupės metodas [10].

4.3.1. Nuoseklusis SOM tinklo ir Sammono algoritmo junginys

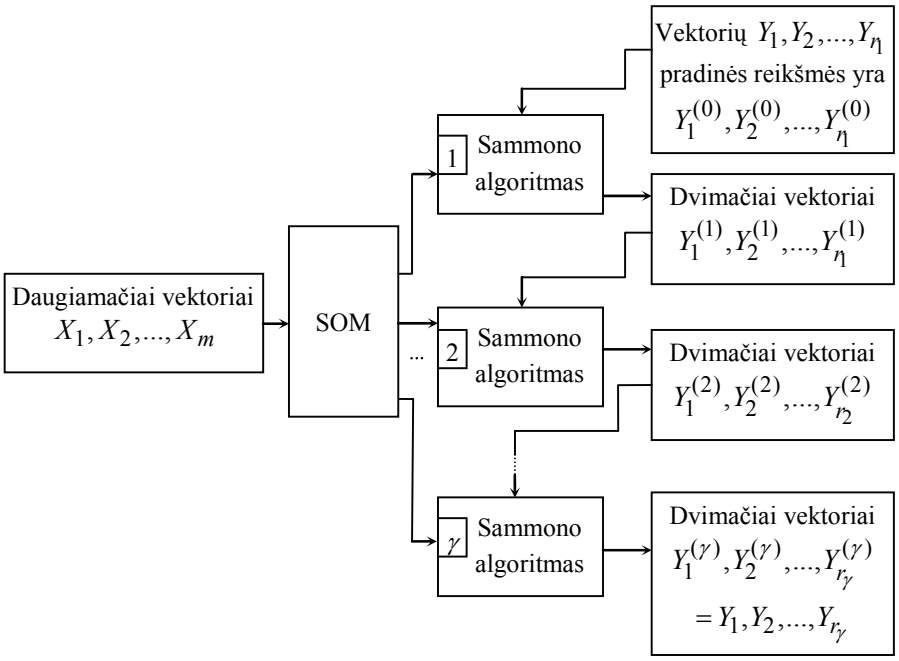
Netuščius SOM žemėlapio langelius atitinkančius vektorius nugalėtojus galima analizuoti Sammono algoritmu. Nuoseklaus šių metodų junginio pradžioje įėjimo vektoriai $X_1, X_2, \ldots, X_m$ pateikiami į SOM tinklą, po SOM mokymo gauti neuronai nugalėtojai Sammono algoritmu atvaizduojami plokštumoje, gaunami dvimačiai vektoriai $Y_1, Y_2, \ldots, Y_m$ (žr. 4.17 pav.) [37]. Neuronų nugalėtojų skaičius *r* paprastai būna mažesnis nei *m*, o tada tarp vektorių $Y_1, Y_2, \ldots, Y_m$ būna ir sutampančių, t. y. kelis vektorius iš $X_1, X_2, \ldots, X_m$ atitinka vienas plokštumos taškas.



4.17 pav. Nuosekliojo SOM tinklo ir Sammono algoritmo junginio schema

4.3.2. Integruotasis SOM tinklo ir Sammono algoritmo junginys

Integruotame SOM tinklo ir Sammono algoritmo junginyje daugiamačiai vektoriai analizuojami Sammono algoritmu atsižvelgiant į saviorganizuojančio neuroninio tinklo mokymosi rezultatus (žr. 4.18 pav.) [44].



4.18 pav. Integruotojo SOM tinklo ir Sammono algoritmo junginio schema

Integruoto SOM tinklo ir Sammono projekcijos junginio algoritmas:

1. Tegul mokymo aibę sudaro *n*-mačiai vektoriai $X_1, X_2, \ldots, X_m$. Neuroninis tinklas bus mokamas naudojant *e* mokymo epochų (viena epocha – tai SOM tinklo mokymo proceso dalis, kai visi vektoriai tam tikra tvarka pateikiami į tinklą po vieną kartą).

2. Mokymo procesas, susidedantis iš e epochų, suskaidomas į lygius mokymo proceso blokus. Prieš neuroninio tinklo mokymą pasirenkama, į kelis tokius blokus γ bus skaidomas visas mokymo procesas. Tegul kiekvieną mokymo proceso bloką sudaro ν mokymo epochų ($e = \nu\gamma$). Raide q pažymimas mokymo bloko, sudaryto iš ν epochų, numeris ($q = 1,\ldots,\gamma$).
3. Vektoriai nugalėtojai, gauti q-ajame mokymo bloke, pažymimi $M_1^{(q)}, M_2^{(q)},\ldots, M_{r_q}^{(q)}$. Jų dvimatės projekcijos, apskaičiuotos Sammono algoritmu, yra:

$$Y_1^{(q)}, Y_2^{(q)},\ldots, Y_{r_q}^{(q)}, \quad Y_i^{(q)} = \left(y_{i1}^{(q)}, y_{i2}^{(q)}\right),\ i = 1,\ldots, r_q.$$

Vektorių nugalėtojų skaičius r_q bus mažesnis arba lygus m. Vektoriai nugalėtojai $M_1^{(1)}, M_2^{(1)},\ldots, M_{r_1}^{(1)}$, gauti po pirmojo mokymo proceso bloko ($q = 1$), yra analizuojami Sammono algoritmu. Kiekvieną iš šių vektorių nugalėtojų atitinka vienas ar keli mokymo aibės $\{X_1, X_2,\ldots, X_m\}$ vektoriai. Pradinės dvimačių vektorių

$$Y_i^{(0)} = \left(y_{i1}^{(0)}, y_{i2}^{(0)}\right), \quad i = 1,\ldots, r_1,$$

komponenčių reikšmės imamos tokios:

$$y_{i1}^{(0)} = i + \frac{1}{3}, \qquad y_{i2}^{(0)} = i + \frac{2}{3}.$$

Naudojantis Sammono algoritmu, apskaičiuojamos vektorių nugalėtojų dvimatės projekcijos $Y_1^{(1)}, Y_2^{(1)},\ldots, Y_{r_1}^{(1)}$.

4. Neuroninio tinklo mokymas tęsiamas toliau. Vektoriai nugalėtojai, gauti po q-ojo mokymo proceso bloko, yra analizuojami Sammono algoritmu. Pradinės dvimačių vektorių $Y_1^{(q)}, Y_2^{(q)},\ldots, Y_{r_q}^{(q)}$ komponentės yra nustatomos atsižvelgiant į ($q-1$)-ajame bloke gautus rezultatus. Bendruoju atveju $r_q \neq r_{q-1}$. Vienas iš galimų dvimačių vektorių pradinių komponenčių reikšmių parinkimo būdų yra toks:
   - Nustatomos kiekvieno dvimačio vektoriaus $Y_i^{(q)}$, atitinkančio neuroną nugalėtoją $M_i^{(q)}$, $i = 1,\ldots, r_q$, komponenčių pradinės reikšmės.

- Randama, kurie mokymo aibės $\{X_1, X_2, \ldots, X_m\}$ vektoriai yra susiję su neuronu nugalėtoju $M_i^{(q)}$. Tegul tai būna vektoriai
$$X_{i_1}, X_{i_2}, \ldots \Big| \left(X_{i_1}, X_{i_2}, \ldots \in \{X_1, X_2, \ldots, X_m\} \right).$$
- Išsiaiškinama, kurie $(q-1)$-ame mokymo proceso bloke gauti neuronai nugalėtojai yra susiję su $X_{i_1}, X_{i_2}, \ldots$. Tegul tai būna neuronai nugalėtojai
$$M_{j_1}^{(q-1)}, M_{j_2}^{(q-1)}, \ldots \Big| \left(M_{j_1}^{(q-1)}, M_{j_2}^{(q-1)}, \ldots \in \left\{ M_1^{(q-1)}, M_2^{(q-1)}, \ldots, M_{r_{q-1}}^{(q-1)} \right\} \right),$$
o jų dvimatės projekcijos, gautos Sammono algoritmu, –
$$Y_{j_1}^{(q-1)}, Y_{j_2}^{(q-1)}, \ldots \Big| \left(Y_{j_1}^{(q-1)}, Y_{j_2}^{(q-1)}, \ldots \in \left\{ Y_1^{(q-1)}, Y_2^{(q-1)}, \ldots, Y_{r_{q-1}}^{(q-1)} \right\} \right).$$
- Pradinės vektoriaus $Y_i^{(q)}$ komponentės imamos lygios vektorių $\{Y_{j_1}^{(q-1)}, Y_{j_2}^{(q-1)}, \ldots\}$ komponenčių vidurkiams.
- Sammono algoritmu skaičiuojamos vektorių nugalėtojų dvimatės projekcijos
$$Y_1^{(q)}, Y_2^{(q)}, \ldots, Y_{r_q}^{(q)}, \quad Y_i^{(q)} = \left(y_{i1}^{(q)}, y_{i2}^{(q)} \right), \; i = 1, \ldots, r_q .$$

5. Neuroninis tinklas mokomas tol, kol $q = \gamma$. Po γ bloko gaunamos *n*-mačių vektorių nugalėtojų $M_1^{(\gamma)}, M_2^{(\gamma)}, \ldots, M_{r_\gamma}^{(\gamma)}$ dvimatės projekcijos $Y_1^{(\gamma)}, Y_2^{(\gamma)}, \ldots, Y_{r_\gamma}^{(\gamma)}$, kurios atitinka *n*-mačių vektorių $X_1, X_2, \ldots, X_m$ dvimates projekcijas.

SOM tinklo ir Sammono algoritmo junginio praktinių taikymų pavyzdžiai aprašyti 5 skyriuje. Ten atskleisti junginio privalumai, palyginti su SOM tinklo ar Sammono algoritmo pavienio naudojimo atvejais.

4.4. Kreivinių komponenčių analizė

SOM tinklas transformuoja turimus duomenis į tam tikrą fiksuotą topologinę struktūrą. Jeigu ši struktūra nesutampa su tikrąja analizuojamų duomenų struktūra, tai gautas atvaizdavimas nėra tikslus. *Kreivinių komponenčių analizė* (KKA, *curvilinear components analysis*, CCA) yra metodas, siejantis SOM tinklo idėją ir netiesinį projekcijos metodą (DS tipo) [27]. Čia išėjimo erdvė yra

tolydi, todėl šiuo metodu gaunamų rezultatų struktūra labiau atitinka tikrąją analizuojamų duomenų struktūrą.

KKA tinklą sudaro du sluoksniai: pirmas sluoksnis atlieka duomenų aibės vektorių kvantavimą, antrasis, vadinamasis projekcijos sluoksnis, atlieka vektorių, gautų iš vektorių kvantavimo sluoksnio, atvaizdavimą. *Vektorių kvantavimas* – tai procesas, kurio metu *n*-mačiai įėjimo vektoriai yra pakeičiami mažesniu kiekiu *n*-mačių išėjimo vektorių.

Po mokymo tinklas turi galimybę rasti naujo vektoriaus, kurio nebuvo kvantuojamų vektorių aibėje, projekciją. Naudojantis KKA galima ir atvirkštinė projekcija, t. y. iš dvimatės erdvės į *n*-matę erdvę sukeičiant įėjimo ir išėjimo sluoksnius.

Kreivinių komponenčių analizės algoritmą sudaro du žingsniai:

1) *n*-mačiai vektoriai $X_1, X_2, \ldots, X_m$ kvantuojami į *s* pradinės erdvės poerdvių (pirmas tinklo sluoksnis) bet kuriuo vektorių kvantavimo metodu [2] ir gaunami *n*-mačiai vektoriai $M_1, M_2, \ldots, M_s$, $s \le m$;
2) atliekama vektorių $M_1, M_2, \ldots, M_s$ netiesinė projekcija (antras tinklo sluoksnis), siekiama minimizuoti skirtumus tarp kvantavimo erdvės ir vaizdo (projekcinės) erdvės vektorių atstumų.

KKA metodu ieškoma mažesnio matmenų skaičiaus vektorių Y_i, $i = 1, \ldots, s$, minimizuojant funkciją

$$E_{KKA} = \frac{1}{2} \sum_{\substack{i,j=1 \\ i \neq j}}^{s} \left(d(M_i, M_j) - d(Y_i, Y_j) \right)^2 F\left(d(Y_i, Y_j), \lambda_y \right),$$

čia $d(M_i, M_j)$ yra atstumas tarp daugiamačių vektorių M_i ir M_j, $d(Y_i, Y_j)$ – atstumas tarp vektorius M_i ir M_j atitinkančių mažesnio matmenų skaičiaus vektorių Y_i ir Y_j, $i, j = 1, \ldots, s$.

$F(d(Y_i, Y_j), \lambda_y)$ yra svorio funkcija, kurios reikšmės yra intervale [0; 1] arba (0; 1). Ji turi būti aprėžta ir monotoniškai mažėjanti.

Svorio funkcija gali būti:

- slenkstinė

$$F\left(d(Y_i, Y_j), \lambda_y \right) = \begin{cases} 1, & \text{jei } d(Y_i, Y_j) \le \lambda_y, \\ 0, & \text{jei } d(Y_i, Y_j) > \lambda_y; \end{cases}$$

- eksponentinė

$$F\left(d(Y_i,Y_j),\lambda_y\right)=\mathrm{e}^{-\lambda_y d(Y_i,Y_j)};$$

- sigmoidinė

$$F\left(d(Y_i,Y_j),\lambda_y\right)=\frac{1}{1+\mathrm{e}^{\lambda_y d(Y_i,Y_j)}}.$$

Parametro λ_y reikšmė yra teigiama, laisvai parenkama.

Kreivinių komponenčių analizėje didžiausias dėmesys skiriamas vektoriams, kurie mažiausiai nutolę vienas nuo kito. Būtent dėl įvestos svorio funkcijos KKA metodas skiriasi nuo įprastinių DS tipo metodų.

Minimizuojant paklaidos funkciją E_{KKA}, dvimačių vektorių $Y_i \in R^2$ komponentės y_{ik}, $i=1,\ldots,s$, $i\neq j$, $k=1,2$, randamos pagal iteracinę formulę [27]

$$y_{ik}(t+1)=y_{ik}(t)+\eta(t)F\left(d(Y_i,Y_j),\lambda_y\right)\times$$

$$\times\left(d(X_i,X_j)-d(Y_i,Y_j)\right)\frac{y_{ik}-y_{jk}}{d(Y_i,Y_j)},\quad \forall i\neq j,$$

čia t yra iteracijos numeris, $\eta(t)=\frac{\eta_0}{1+t}$ – mokymo parametras, priklausantis nuo t, j yra fiksuojamas kiekvienoje iteracijoje (parenkamas atsitiktinai), η_0 – laisvai parenkamas parametras.

Kreivinių atstumų analizės (*curvilinear distance analysis*, CDA) metodas yra labai panašus į kreivinių komponenčių analizės metodą [98], [99]. Tik čia naudojami ne Euklido atstumai, o geodeziniai (kreiviniai) atstumai pradinėje daugiamatėje erdvėje (kaip ir ISOMAP metode, aprašytame 2.2.2 skyrelyje). Kaip ir kreivinių komponenčių metode, pagrindinis dėmesys skiriamas atstumams tarp tų taškų, kurie yra mažiausiai nutolę vienas nuo kito.

4.5. Daugiamatės skalės taikant dirbtinius neuroninius tinklus

Šiame skyrelyje apžvelgiami du dirbtinių neuroninių tinklų taikymai daugiamačių skalių metodo (DS) grupės projekcijoms rasti. Analizuojamos dvi strategijos:

1) kai DS tipo projekcija yra randama taikant tiesioginio sklidimo neuroninį tinklą, mokomą įprastiniu „klaidos skleidimo atgal“ algoritmu (mokymas su mokytoju) [29];

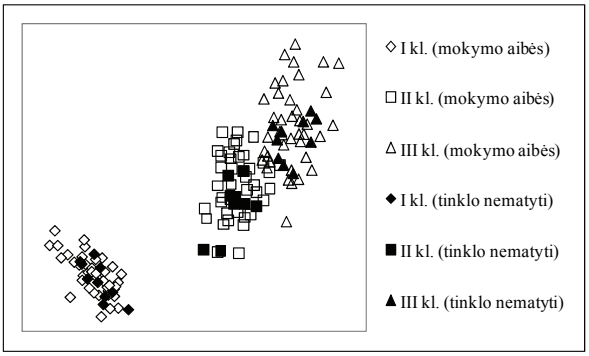
2) kai DS tipo projekcijai rasti yra naudojamas neuroninis tinklas, mokomas tam tikru „klaidos skleidimo atgal“ algoritmu, pavadintu SAMANN (mokymas be mokytojo) [106].

4.5.1. Mokymo su mokytoju strategija

Sammono algoritmo vienas iš trūkumų yra tai, kad norint atvaizduoti plokštumoje analizuojamų vektorių aibėje atsiradusį naują vektorių (tašką), reikia perskaičiuoti visų jau atvaizduotų vektorių (taškų) projekcijas. Vienas iš būdų išvengti šio trūkumo – naudoti dirbtinius neuroninius tinklus.

Tegul turime *n*-mačius taškus $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $i = 1, \ldots, m$. Taškai X_k, $k = 1, \ldots, \overline{m}$, $\overline{m} < m$, Sammono algoritmu arba kitu DS tipo metodu atvaizduojami plokštumoje, t. y. gaunamos taškų X_k projekcijos $Y_k = (y_{k1}, y_{k2})$. Vėliau taškų X_k, $k = 1, \ldots, \overline{m}$, komponentės $x_{k1}, x_{k2}, \ldots, x_{kn}$ pateikiamos į tiesioginio sklidimo neuroninį tinklą kaip įėjimai. Norimos išėjimų reikšmės imamos taškų X_k projekcijų Y_k komponentės y_{k1}, y_{k2}. Tinklas mokomas įprastiniu „klaidos skleidimo atgal“ algoritmu. Tada tinklo dar „nematyti“, t. y. tinklo mokymui nenaudoti taškai X_l, $l = \overline{m}+1, \ldots, m$, pateikiami į jau išmokytą tinklą, išėjimuose gaunamos jų projekcijų Y_l komponentės [1], [29].

Irisų duomenys, vizualizuoti Sammono algoritmu naudojant neuroninį tinklą, mokomą įprastiniu „klaidos skleidimo atgal“ algoritmu, pateikti 4.19 paveiksle. Iš pradžių 120 keturmačių vektorių (po 40 vektorių iš kiekvienos irisų klasės) Sammono algoritmu atvaizduojami plokštumoje, t. y. randami dvimačiai vektoriai.



4.19 pav. Sammono projekcija naudojant DNT (vizualizuoti irisų duomenys)

Neuroninio tinklo mokymo aibę sudaro šie 120 vektorių: keturmačiai vektoriai kaip įėjimai, dvimačiai – kaip išėjimai. Tada į išmokytą tinklą pateikiami likusieji 30 vektorių (po 10 iš kiekvienos klasės), randamos jų projekcijos. Naudotasi vieno paslėptojo sluoksnio neuroniniu tinklu. Paslėptajame sluoksnyje yra 10 neuronų. Iš paveikslo akivaizdu, kad tinklo „nematyti“ taškai plokštumoje randa vietą tarp savo klasės taškų.

4.5.2. Mokymo be mokytojo strategija (SAMANN)

Daugiamačių skalių tipo projekcijai rasti yra naudojamas neuroninis tinklas, kuris mokomas specifiniu „klaidos skleidimo atgal“ algoritmu, pavadintu SAMANN (mokymas be mokytojo) [106]. Detalūs metodo tyrimai pateikti [78], [111] ir [112] darbuose.

J. Mao ir A. K. Jainas 1995 metais pasiūlė Sammono paklaidą minimizuoti naudojant tiesioginio sklidimo neuroninius tinklus, mokomus pagal „klaidos skleidimo atgal“ mokymo taisyklę. Pagal ją įprastas tiesioginio sklidimo neuroninis tinklas (žr. 4.20 pav.) gali realizuoti Sammono projekciją mokymo be mokytojo būdu.

Analizuojamų duomenų aibę sudaro m n-mačių vektorių $X = \{X_1, X_2, \ldots, X_m\}$. Jų komponenčių reikšmės naudojamos tinklo mokymui kaip įėjimų reikšmės. Kiekviename mokymo žingsnyje į neuroninį tinklą pateikiami du n-mačiai vektoriai $X_\mu = (x_{\mu 1}, x_{\mu 2}, \ldots, x_{\mu n})$ ir $X_\nu = (x_{\nu 1}, x_{\nu 2}, \ldots, x_{\nu n})$, išėjimuose siekiama gauti jų projekcijas d-matėje erdvėje, t. y. vektorius $Y_\mu = (y_{\mu 1}, y_{\mu 2}, \ldots, y_{\mu d})$ ir $Y_\nu = (y_{\nu 1}, y_{\nu 2}, \ldots, y_{\nu d})$ $d < n$.

Sammono projekcijos paklaida E_S skaičiuojama pagal formulę

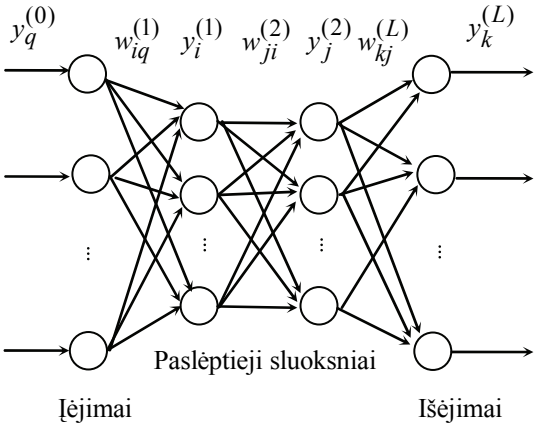
$$E_S = \frac{1}{\sum_{\mu=1}^{m-1}\sum_{\nu=\mu+1}^{m} d(X_\mu, X_\nu)} \sum_{\mu=1}^{m-1}\sum_{\nu=\mu+1}^{m} \frac{[d(X_\mu, X_\nu) - d(Y_\mu, Y_\nu)]^2}{d(X_\mu, X_\nu)}, \qquad (4.16)$$

čia $d(X_\mu, X_\nu)$ yra atstumas tarp n-mačių vektorių X_μ ir X_ν, $d(Y_\mu, Y_\nu)$ – atstumas tarp juos atitinkančių d-mačių vektorių Y_μ ir Y_ν, $d < n$, m – analizuojamų vektorių skaičius.

Metodo idėja yra ta, kad 4.20 paveiksle pavaizduotam tinklui pateikiami vienas paskui kitą du n-mačiai vektoriai X_μ ir X_ν, apskaičiuojami neuroninio tinklo atitinkami išėjimai Y_μ ir Y_ν, skaičiuojamas atstumas tarp vektorių Y_μ ir

Y_ν ir projekcijos paklaidos E_S, apibrėžtos (4.16) formule, reikšmė. Atsižvelgiant į ją, keičiami neuronų svoriai.

Tegul $y_j^{(l)}$, $j = 1,\ldots,n_l$, $l = 0,\ldots,L$, yra *j*-ojo neurono išėjimas *l*-ajame sluoksnyje, čia n_l yra *l*-ojo sluoksnio neuronų skaičius, *L* – neuronų sluoksnių skaičius.



4.20 pav. Tiesioginio sklidimo dviejų paslėptųjų sluoksnių neuroninis tinklas
Sammono projekcijai

Neuroninio tinklo įėjimai – tai įėjimo vektoriaus komponentės $y_q^{(0)} = x_q$, $q = 1,\ldots,n$. Jungties tarp *i*-ojo neurono $(l-1)$-ajame sluoksnyje ir *j*-ojo neurono *l*-ajame sluoksnyje svorį žymi $w_{ji}^{(l)}$, *j*-ojo neurono *l*-ajame sluoksnyje slenksčio reikšmę (*bias*) nurodo $w_{j0}^{(l)}$, o $y_0^{(l)} = 1$.

Kiekvieno neurono išėjimo reikšmei skaičiuoti naudojama sigmoidinė funkcija

$$f(a) = \frac{1}{1+\mathrm{e}^{-a}},$$

kurios reikšmių intervalas yra (0; 1), čia *a* – visų neuronų įėjimų ir jų svorių sandaugų suma. Ši funkcija yra patogi tuo, kad jos išvestinė pagal *a* priklauso tik nuo jos pačios reikšmės:

$$f'(a) = [1 - f(a)]f(a),$$

Taigi *j*-ojo neurono *l*-ajame sluoksnyje išėjimas išreiškiamas taip:

$$y_j^{(l)} = f\left(\sum_{i=0}^{n_{l-1}} w_{ji}^{(l)} y_i^{(l-1)}\right), \quad l = 1,\ldots,L .$$

Euklido atstumas $d(Y_\mu, Y_\nu)$ tarp dviejų *d*-mačių vektorių Y_μ ir Y_ν apskaičiuojamas pagal formulę

$$d(Y_\mu, Y_\nu) = \left\{\sum_{k=1}^{d}\left[y_{\mu k}^{(L)} - y_{\nu k}^{(L)}\right]^2\right\}^{\frac{1}{2}},$$

čia $y_{\mu k}^{(L)}$ ($y_{\nu k}^{(L)}$) yra vektoriaus Y_μ (Y_ν) *k*-oji komponentė *L*-ajame, t. y. išėjimų, sluoksnyje.

Atkreipkime dėmesį į tai, kad dėl sigmoidinės funkcijos reikšmių intervalo, kiekvieno išėjimo intervalas yra (0; 1). Jei įėjimo reikšmių intervalas yra platus, taikant projekcijos algoritmą, neįmanoma išlaikyti santykinių atstumų tarp analizuojamų vektorių. Todėl visi įėjimo vektoriai normuojami norint suvienodinti atstumus tarp vektorių pradinėje erdvėje ir vaizdo (projekcijos) erdvėje (detaliau apie normavimą skaitykite 1.3 skyrelyje).

Pažymėkime

$$\lambda = \frac{1}{\sum_{\mu=1}^{m-1}\sum_{\nu=\mu+1}^{m} d(X_\mu, X_\nu)} .$$

Dydžio λ reikšmė nepriklauso nuo tinklo, taigi ją galima apskaičiuoti iš anksto. Tuomet Sammono paklaidą taškų X_μ ir X_ν porai galima rasti pagal formulę

$$E_{\mu\nu} = \lambda \frac{\left[d(X_\mu, X_\nu) - d(Y_\mu, Y_\nu)\right]^2}{d(X_\mu, X_\nu)},$$

o visų taškų atveju

$$E_S = \sum_{\mu=1}^{m-1}\sum_{\nu=\mu+1}^{m} E_{\mu\nu} .$$

Dydis $E_{\mu\nu}$ yra proporcingas atstumų tarp vektorių X_μ ir X_ν bei tarp Y_μ ir Y_ν skirtumui, todėl jis labiau tinkamas svorių keitimo taisyklėms nustatyti.

J. Mao ir A. K. Jainas daugiasluoksniam tiesioginio sklidimo tinklui pasiūlė svorių keitimo taisyklę, pagrįstą gradientinio nusileidimo metodu. Ja naudojantis, minimizuojama Sammono projekcijos paklaida E_S.

Išėjimo sluoksnyje $(l = L)$

$$\frac{\partial E_{\mu\nu}}{\partial w_{kj}^{(L)}} = \left(\frac{\partial E_{\mu\nu}}{\partial d(Y_\mu, Y_\nu)}\right)\left(\frac{\partial d(Y_\mu, Y_\nu)}{\partial\left[y_{\mu k}^{(L)} - y_{\nu k}^{(L)}\right]}\right)\left(\frac{\partial\left[y_{\mu k}^{(L)} - y_{\nu k}^{(L)}\right]}{\partial w_{kj}^{(L)}}\right) =$$

$$= \left(-2\lambda\frac{d(X_\mu, X_\nu) - \partial d(Y_\mu, Y_\nu)}{d(X_\mu, X_\nu)}\right)\left(\frac{y_{\mu k}^{(L)} - y_{\nu k}^{(L)}}{d_{\mu\nu}}\right)\times \tag{4.17}$$

$$\times\left(f'\left(a_{\mu k}^{(L)}\right)y_{\mu j}^{(L-1)} - f'\left(a_{\nu k}^{(L)}\right)y_{\nu j}^{(L-1)}\right),$$

čia $f'(a_{\mu k}^{(L)})$ ir $f'(a_{\nu k}^{(L)})$ yra k-ojo elemento L-ajame (išėjimo) sluoksnyje sigmoidinės funkcijos išvestinių pagal $a_{\mu k}$ ir $a_{\nu k}$ reikšmės:

$$f'(a_{\mu k}^{(L)}) = \left(1 - y_{\mu k}^{(L)}\right)y_{\mu k}^{(L)}, \tag{4.18}$$

$$f'(a_{\nu k}^{(L)}) = \left(1 - y_{\nu k}^{(L)}\right)y_{\nu k}^{(L)}.$$

Tegul

$$\delta_k^{(L)}(\mu,\nu) = -2\lambda\frac{d(X_\mu, X_\nu) - d(Y_\mu, Y_\nu)}{d(X_\mu, X_\nu)d(Y_\mu, Y_\nu)}\left(y_{\mu k}^{(L)} - y_{\nu k}^{(L)}\right), \tag{4.19}$$

$$\Delta_{kj}^{(L)}(\mu) = \delta_k^{(L)}(\mu,\nu)\left(1 - y_{\mu k}^{(L)}\right)y_{\mu k}^{(L)}, \tag{4.20}$$

$$\Delta_{kj}^{(L)}(\nu) = \delta_k^{(L)}(\mu,\nu)\left(1 - y_{\nu k}^{(L)}\right)y_{\nu k}^{(L)}. \tag{4.21}$$

Įstatę (4.18)–(4.21) formules į (4.17), gauname:

$$\frac{\partial E_{\mu\nu}}{\partial w_{kj}^{(L)}} = \Delta_{kj}^{(L)}(\mu)y_{\mu j}^{(L-1)} - \Delta_{kj}^{(L)}(\nu)y_{\nu j}^{(L-1)}.$$

Taigi išėjimo sluoksnio svorių keitimo taisyklė yra tokia:

$$\Delta w_{kj}^{(L)} = -\eta \frac{\partial E_{\mu\nu}}{\partial w_{kj}^{(L)}} = -\eta(\Delta_{kj}^{(L)}(\mu) y_{\mu j}^{L-1} - \Delta_{kj}^{(L)}(\nu) y_{\nu j}^{L-1}), \tag{4.22}$$

čia η yra teigiamas daugiklis, kuriuo reguliuojamas gradientinio optimizavimo žingsnio ilgis. Panašiai gaunamos bendros svorių keitimo taisyklės kiekvienam paslėptajam sluoksniui $l = 1,\ldots,L-1$:

$$\Delta w_{ji}^{(l)} = -\eta \frac{\partial E_{\mu\nu}}{\partial w_{ji}^{(l)}} = -\eta(\Delta_{ji}^{(l)}(\mu) y_{\mu i}^{l-1} - \Delta_{ji}^{(l)}(\nu) y_{\nu i}^{l-1}), \tag{4.23}$$

čia

$$\Delta_{ji}^{(l)}(\mu) = \delta_j^{(l)}(\mu)\left(1 - y_{\mu j}^{(l)}\right) y_{\mu j}^{(l)}, \tag{4.24}$$

$$\Delta_{ji}^{(l)}(\nu) = \delta_j^{(l)}(\nu)\left(1 - y_{\nu j}^{(l)}\right) y_{\nu j}^{(l)}, \tag{4.25}$$

$$\delta_j^{(l)}(\mu) = \sum_{k=1}^{d} \Delta_{kj}^{(l+1)}(\mu) w_{kj}^{(l+1)}, \tag{4.26}$$

$$\delta_j^{(l)}(\nu) = \sum_{k=1}^{d} \Delta_{kj}^{(l+1)}(\nu) w_{kj}^{(l+1)}. \tag{4.27}$$

Kaip ir standartiniame „klaidos skleidimo atgal“ mokymo algoritme, dydžiai $\delta_j^{(l)}(\mu)$ ir $\delta_j^{(l)}(\nu)$ keičiasi grįžtant iš $(l+1)$-ojo sluoksnio į l-ąjį sluoksnį.

Iš (4.22) ir (4.23) formulių išplaukia, kad norint atnaujinti neuroninio tinklo svorius, į tinklą tuo pat metu reikia teikti vektorių porą. Tai galima padaryti sukonstravus du identiškus tinklus arba tiesiog laikant atmintyje visus pirmojo vektoriaus išėjimus prieš teikiant antrąjį vektorių.

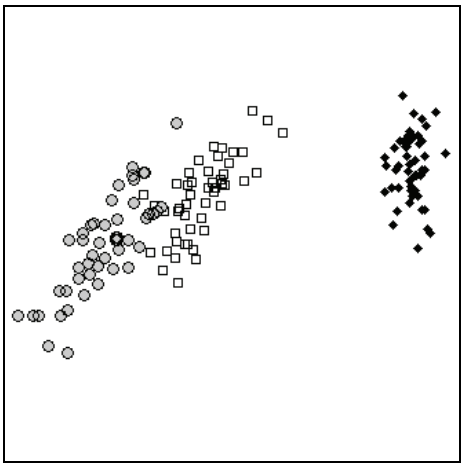
Apibendrinta algoritmo SAMMAN schema yra tokia:

1. Generuojamos atsitiktinės pradinės SAMANN tinklo svorių reikšmės.
2. Atsitiktinai parenkama *n*-mačių vektorių pora, kuri yra pateikiama tinklui ir apskaičiuojami tinklo parametrai.

3. Svoriai keičiami pagal (4.22) ir (4.23) formules „klaidos skleidimo atgal“ būdu, pradedant nuo išėjimo sluoksnio.
4. Kelis kartus kartojami 2–3 žingsniai ir tinklas išmokomas.
5. Pateikiami visi vektoriai ir apskaičiuojami tinklo išėjimai. Skaičiuojama Sammono paklaida. Jeigu jos reikšmė mažesnė už pasirinktą slenkstį arba iteracijų skaičius viršija nustatytąjį, tuomet algoritmas sustabdomas, priešingu atveju vėl pradedama nuo 2 žingsnio.

Algoritmu SAMANN gautas irisų išsidėstymas plokštumoje pavaizduotas 4.21 paveiksle. Rezultatas yra labai panašus į gaunamus kitais netiesinės projekcijos metodais, nes tinklą siekiama apmokyti taip, kad būtų minimizuojama projekcijos paklaida E_S, kuri yra vienas iš dažnai naudojamų vizualizavimo kokybės kriterijų.

Algoritmo trūkumas yra tai, kad ilgai trunka neuroninio tinklo mokymas. Tačiau šis algoritmas turi vieną išskirtinę savybę – galimybę iškart atvaizduoti naują n-matį tašką X_{m+1} neperskaičiuojant tinklo svorių. Taigi jeigu analizuojamų vektorių aibėje atsiranda naujas n-matis taškas X_{m+1} ir jis pateikiamas jau išmokytam SAMANN tinklui, tai tinklo išėjime gaunamos taško Y_{m+1}, kuris yra X_{m+1} projekcija, koordinatės. Žinoma, jeigu tų naujų taškų yra daug, po tam tikro laiko tinklą reikia mokyti iš naujo ir rasti naujas svorių reikšmes.

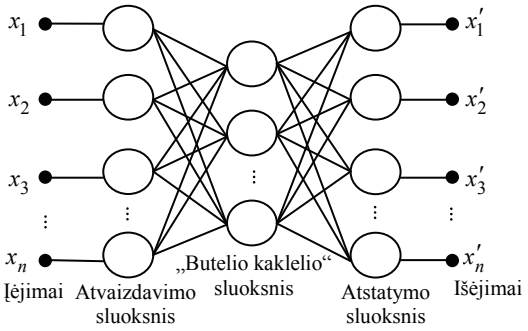


4.21 pav. Algoritmu SAMMAN vizualizuoti irisų duomenys

4.6. Autoasociatyvieji neuroniniai tinklai

Autoasociatyvieji neuroniniai tinklai (*autoassociative neural network*) [5], [91] dar dažnai vadinami *autokoderių tinklais* [28], [72]. Jie naudojami matmenų skaičiui mažinti išskiriant *d* neuronų iš vadinamojo „butelio kaklelio“ (*bottleneck*) sluoksnio, sudaryto iš mažiau elementų nei įėjimo ir išėjimo sluoksniai, čia *d* yra vaizdo erdvės matmenų skaičius.

Autoasociatyvusis neuroninis tinklas sudarytas iš dviejų dalių: pirma dalis transformuoja pradinius analizuojamus daugiamačius duomenis į mažesnio skaičiaus matmenų erdvę (atvaizdavimo sluoksnis), o antroji – rekonstruoja (atstato) pradinius duomenis iš gautų projekcijų (atstatymo sluoksnis). Tai pavaizduota 4.22 paveiksle.



4.22 pav. Autoasociatyviojo neuroninio tinklo schema

Tinklo mokymo proceso metu yra minimizuojama vidutinė kvadratinė paklaida E_{aaNT}, gaunama tarp tinklo įėjimo vektoriaus $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$ ir išėjimo vektoriaus $X'_i = (x'_{i1}, x'_{i2}, \ldots, x'_{in})$, $i = 1, \ldots, m$, atitinkamų komponenčių reikšmių:

$$E_{aaNT} = \sum_{i=1}^{m} \sum_{j=1}^{n} (x_{ij} - x'_{ij})^2 .$$

Neuroninis tinklas mokomas duomenų vektoriais X_i, o tinklo vidurinio paslėptojo „butelio kaklelio“ sluoksnio neuronų išėjimuose gaunamos analizuojamų duomenų projekcijos Y_i *d*-matėje erdvėje, išlaikant tikslią pradinių duomenų struktūrą.

4.7. Neuroninių skalių metodas

Neuroninių skalių metodas (*NeuroScale*) [138] remiasi radialinių bazinių funkcijų (RBF) neuroniniais tinklais [103], [104]. Tegul turime *n*-mačius vektorius $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $i = 1, \ldots, m$. Norime rasti jų projekcijas $Y_i = (y_{i1}, y_{i2}, ..., y_{id})$ mažesnio skaičiaus matmenų erdvėje R^d, $d < n$. Naudojantis RBF neuroniniu tinklu, galima nustatyti vektorių Y_i komponenčių reikšmes $y_{i1}, y_{i2}, \ldots, y_{id}$. RBF tinkle yra *n* įėjimų ($x_1, x_2, \ldots, x_n$) ir *d* išėjimų ($y_1, y_2, \ldots, y_d$). Tinklą sudaro bazinių funkcijų sluoksnis ir išėjimų sluoksnis.

Kaip ir visais projekcijos metodais, ieškoma tokių vektorių Y_i, kad būtų minimali paklaida

$$E_{NS} = \sum_{i<j} \left(d(X_i, X_j) - d(Y_i, Y_j) \right)^2 ,$$

čia $d(X_i, X_j)$ yra atstumas tarp vektorių X_i ir X_j, o $d(Y_i, Y_j)$ – atstumas tarp vektorių Y_i ir Y_j, $i, j = 1, \ldots, m$. Taškai Y_i yra generuojami radialinių bazinių funkcijų tinklo, kai įėjimai yra vektoriai X_i, $i = 1, \ldots, m$. Naudojamos tokios formos radialinės bazinės funkcijos:

$$\varphi_k(X_i) = \varphi_k \left(\| X_i - \mu_k \| \right), \quad k = 1, \ldots, s ,$$

t. y. φ_k reikšmė priklauso nuo atstumo tarp vektoriaus X_i ir tam tikro vektoriaus μ_k, vadinamojo radialinės bazinės funkcijos φ_k centro, čia *s* yra bazinių funkcijų skaičius.

Atstumas

$$d(Y_i, Y_j) = \sum_{l=1}^{d} \left(\sum_{k=1}^{s} w_{lk} \left[\phi_k \left(\| X_i - \mu_k \| \right) - \phi_k \left(\| X_j - \mu_k \| \right) \right] \right)^2 ,$$

čia w_{lk} yra svoris jungties tarp *k*-osios bazinės funkcijos ir *l*-ojo išėjimo. Mokant tinklą, t. y. sprendžiant uždavinį

$$\min_{\substack{w_{lk} \\ l=1,\ldots,d \\ k=1,\ldots,s}} E_{NS} ,$$

randamas optimalus Y_i, $i = 1, \ldots, m$, rinkinys.

5. Vizualiosios analizės taikymai

Šiame skyriuje apžvelgiami daugiamačių duomenų vizualiosios analizės praktinių taikymų pavyzdžiai ir gautų rezultatų galimos interpretacijos. Šie taikymai visapusiškai atskleidžia vizualiosios analizės galimybes ir privalumus.

Aprašomus taikymus galima būtų suskirstyti į kelias grupes:

1) taikymai socialiniuose moksluose;
2) taikymai medicinoje ir farmakologijoje;
3) koreliacinių matricų vizualioji analizė.

5.1. Socialiniai duomenys

Šiame skyrelyje pateikiami Centrinės Europos valstybių ekonominės ir socialinės būklės [48], Seimo narių balsavimų [93] ir bendrojo lavinimo mokyklų [42] vizualiosios analizės pavyzdžiai.

5.1.1. Centrinės Europos valstybių ekonominė ir socialinė būklė

Duomenys apie šalių ekonominę-socialinę būklę paimti iš JAV CŽV duomenų bazės *The World Factbook 1999* (http://www.odci.gov/cia/publications/factbook/country.html) ir Pasaulio banko duomenų bazės *Country Data* (http://www.worldbank.org/data/). Parinkti dažnai naudojami esminiai ekonominiai-socialiniai rodikliai [48]:

x_1 – kūdikių mirtingumas (mirtys/1000 gimusių gyvų),

x_2 – bendrasis nacionalinis produktas (BNP), tenkantis vienam gyventojui, išreikštas JAV doleriais vertinant nacionalinės valiutos perkamąją galią, o ne keitimo kursą,

x_3 – BNP procentinė dalis, sukurta industriniame ir aptarnavimo (ne agrariniame) sektoriuose,

x_4 – eksportas tūkstančiais JAV dolerių, tenkantis vienam gyventojui,

x_5 – telefonų skaičius, tenkantis vienam gyventojui.

Lyginta 10 šalių: (1) Vengrija, (2) Čekija, (3) Lietuva, (4) Latvija, (5) Slovakija, (6) Lenkija, (7) Rumunija, (8) Estija, (9) Bulgarija, (10) Slovėnija. Kiekvieną valstybę apibūdina 5-matis ($n = 5$) vektorius $X_i = (x_{i1}, x_{i2}, \ldots, x_{i5})$. Iš šių vektorių sudaroma duomenų aibė

$$X = \{X_1, X_2, \ldots, X_{10}\} = \{x_{ij},\ i = 1, \ldots, 10,\ j = 1, \ldots, 5\},$$

čia indeksas i rodo šalies numerį, o indeksas j – rodiklio (parametro) numerį. Įveskime dar tris papildomus objektus X_{AVE}, X_{MIN}, X_{MAX}, atitinkančius parametrų vidurkių, blogiausių ir geriausių reikšmių rinkinius.

Objektą X_{AVE} apibūdinančių parametrų reikšmės yra visų 10 šalių atitinkamų parametrų reikšmių vidurkiai:

$$x_{\mathrm{AVE}\,j} = \frac{1}{10}\sum_{i=1}^{10} x_{ij}\,.$$

Objektą X_{MIN} nusako parametrų reikšmės, gaunamos išrenkant blogiausias iš visų 10 šalių atitinkamų parametrų reikšmių:

$$x_{\mathrm{MIN}\,j} = \min_{i=1,\ldots,10}(x_{ij})\,.$$

Objektą X_{MAX} apibrėžia parametrų reikšmės, gaunamos išrenkant geriausias iš visų 10 šalių atitinkamų parametrų reikšmių:

$$x_{\mathrm{MAX}\,j} = \max_{i=1,\ldots,10}(x_{ij})\,.$$

Tokie trys dirbtiniai objektai X_{AVE}, X_{MIN}, X_{MAX} bus naudingi mūsų analizei, nes turėsime aiškiai apibrėžtus atskaitos taškus – vidutinį, blogiausią ir geriausią objektus. Analizuosime objektus (vektorius)

$$X_1, X_2, \ldots, X_{10}, X_{\mathrm{AVE}}, X_{\mathrm{MIN}}, X_{\mathrm{MAX}}\,.$$

Parametrai $x_1, x_2, \ldots, x_5$ apibūdina šiuos 13 objektų įvairiais ekonominiais ir socialiniais aspektais. Jų matavimo vienetai yra skirtingi ir nominalios reikšmės skiriasi eilėmis. Todėl prieš analizuojant duomenis, būtina suvienodinti parametrų mastelius. Duomenys sunormuoti taip, kad parametrų vidurkiai būtų lygūs 0, o dispersijos lygios 1 (detaliau apie duomenų normavimą skaitykite 1.3 skyrelyje).

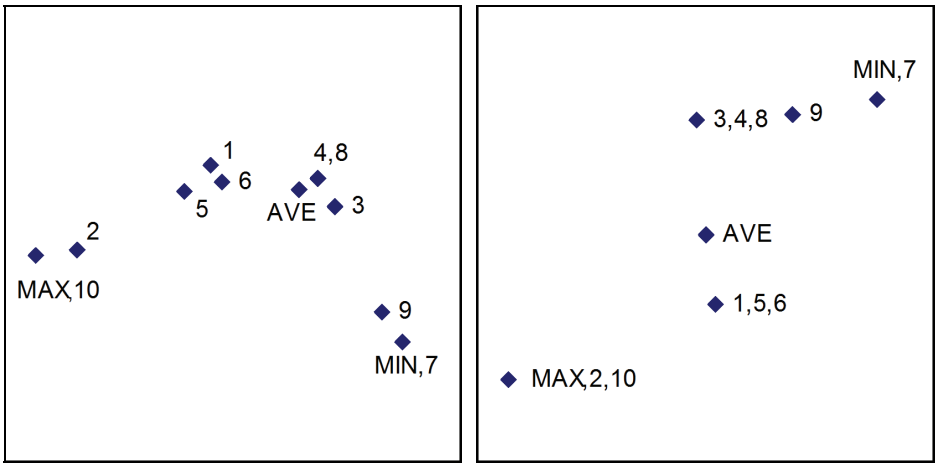
Iš pradžių valstybes apibūdinantys duomenų vektoriai analizuojami naudojantis SOM tinklu (žr. 5.1 ir 5.2 lenteles). Lentelėse skaičiai nurodo analizuojamų objektų (valstybių) numerius, taip pat vidutinį (AVE), blogiausią (MIN) ir geriausią (MAX) objektus. Vien iš jų yra gana sunku nustatyti, kokios „tolimos“ (nepanašios) yra valstybės, esančios gretimuose lentelės langeliuose. Pavyzdžiui, iš 5.1 lentelės matyti, kad 9-oji valstybė yra labai arti vidurkio (AVE). O kaip yra iš tikrųjų? Atvaizduokime plokštumoje SOM tinklo neuronus nugalėtojus taikydami Sammono algoritmą (žr. 5.1 pav.). Matome, kad 9-oji valstybė nėra arti vidurkio. Vidurkiui daug artimesnės valstybės, kurių numeriai yra 1, 5, 6 ir 3, 4, 8.

5.1 lentelė. Centrinės Europos valstybių išsidėstymas SOM [3x3] tinkle

MAX, 2, 10		1, 5, 6
	AVE	
MIN, 7	9	3, 4, 8

5.2 lentelė. Centrinės Europos valstybių išsidėstymas SOM [4x4] tinkle

MAX, 10			MIN, 7
2			9
5			3
1	6	AVE	4, 8



a) b)

5.1 pav. Centrinės Europos valstybių išsidėstymas, gautas SOM ir Sammono algoritmo junginiu: a) SOM [3x3], b) SOM [4x4]

Iš 5.1 ir 5.2 lentelių ir 5.1 paveikslo galime daryti išvadas apie Centrinės Europos valstybių artimumą ekonominiu ir socialiniu aspektais:

- Geriausių valstybių grupę sudaro Slovėnija (10) ir Čekija (2), nes prie jų yra arti objektas MAX; blogiausioje padėtyje yra Rumunija (7) ir Bulgarija (9), nes prie jų yra arti objektas MIN.
- Kitos valstybės yra gana panašios. Jos išsidėsčiusios apie objektą AVE. Tačiau 5.1a paveiksle vizualiai galime išskirti du jų pogrupius: (a) Vengrija (1), Slovakija (5) ir Lenkija (6); (b) Baltijos valstybės – Lietuva (3), Latvija (4) ir Estija (8).

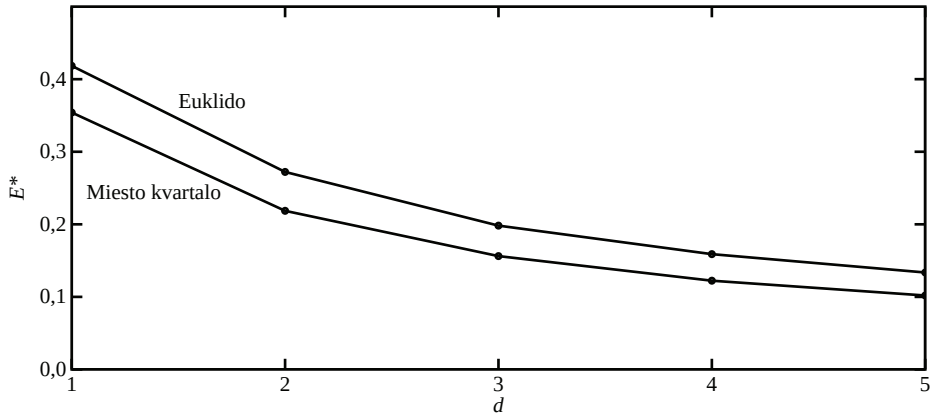
5.1.2. Seimo narių balsavimai

Šiame skyrelyje analizuojami 138 Lietuvos Respublikos Seimo narių 370 balsavimų Seime 2005 ir 2006 metais duomenys. Analizės tikslas – įžvelgti partijų skirtingumus ir panašumas remiantis partijų narių balsavimais [93]. Balsavimas „už“ žymimas „2“, „prieš“ – „–2“, nedalyvavimas balsavime arba susilaikymas – „0“. Šie duomenys yra prieinami internete adresu http://surface.lt/krilaviciust/publications/data.

Seimo narių skirtingumai vertinami skaičiuojant Euklido arba miesto kvartalo atstumus tarp jų balsavimų vektorių. Čia balsavimo vektorių, kuris charakterizuoja atskirą seimo narį, sudaro $n = 370$ komponenčių, kurios įgyja reikšmę –2, 0 arba 2, $m = 138$.

Pirmiausia paanalizuokime šių duomenų daugiamatiškumą. Santykinės daugiamačių skalių paklaidos, apibrėžtos (3.4) formule, priklausomybė nuo vaizdo erdvės matmenų skaičiaus d pavaizduota 5.2 paveiksle. Kaip matyti, santykinė DS paklaida mažėja gana lėtai, gerokai lėčiau negu empirinių duomenų, nagrinėtų 3.5 skyrelyje. Tai rodo, kad duomenys pasklidę visoje erdvėje, o ne mažesnio skaičiaus matmenų poerdvyje. DS su miesto kvartalo atstumais santykinė paklaida yra šiek tiek mažesnė negu DS su Euklido atstumais.

Partijų artimumo analizei, remiantis partijų narių balsavimais, duomenys yra vizualizuojami naudojantis DS. Gautuose vaizduose Seimo nariai pažymėti frakcijos, kuriai priklauso, numeriu.

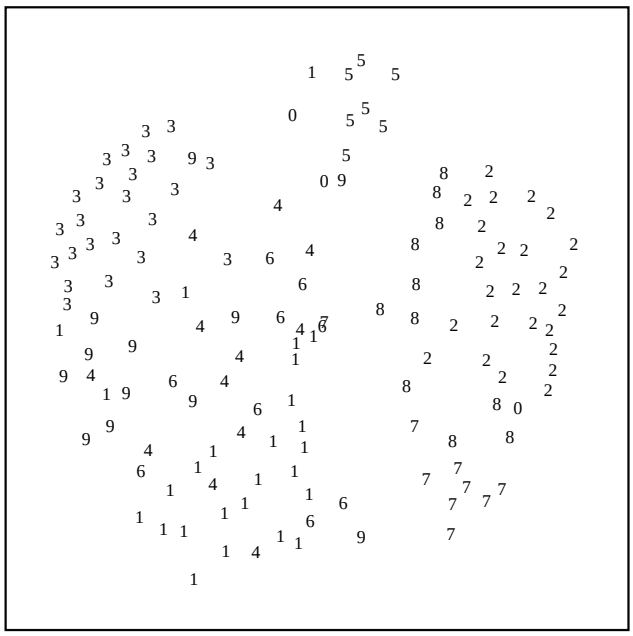


5.2 pav. Seimo narių balsavimo duomenų santykinės DS paklaidos priklausomybė nuo vaizdo erdvės matmenų skaičiaus d

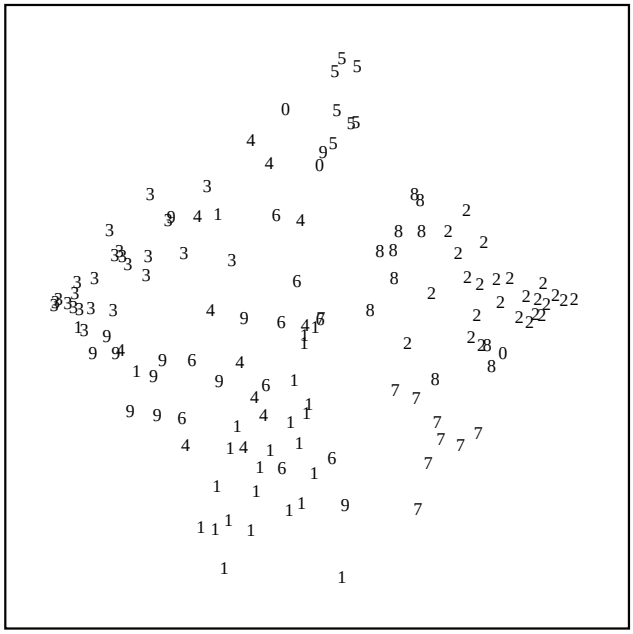
Frakcijų sąrašas:

1 – LSDPF (Lietuvos socialdemokratų partijos frakcija),
2 – TSF (Tėvynės Sąjungos frakcija),
3 – DPF (Darbo partijos frakcija),
4 – VLF (Valstiečių liaudininkų frakcija),
5 – TTLDF (Tvarkos ir teisingumo (liberalų demokratų) frakcija),
6 – NSF (Naujosios sąjungos (socialliberalų) frakcija),
7 – LCSF (Liberalų ir centro sąjungos frakcija),
8 – LF (Liberalų sąjūdžio frakcija),
9 – PDF (Pilietinės demokratijos frakcija),
0 – MG (Mišri Seimo narių grupė).

Daugiamatės skalės yra invariantinės sukimo ir atspindžių atžvilgiu, todėl jos yra pasuktos taip, kad dešiniosios pakraipos partijos būtų vaizduojamos dešinėje, o kairiosios – kairėje. Seimo narių balsavimo duomenų vaizdą DS, gautą naudojantis Euklido atstumais, matome 5.3 paveiksle, o vaizdą DS, gautą naudojantis miesto kvartalo atstumais, – 5.4 paveiksle.



5.3 pav. Seimo narių balsavimo duomenų vaizdas DS, gautas naudojantis Euklido atstumais



5.4 pav. Seimo narių balsavimo duomenų vaizdas DS, gautas naudojantis miesto kvartalo atstumais

Iš paveikslų matyti, kad partijų nariai grupuojami į klasterius. Takoskyra tarp dešiniųjų ir kairiųjų partijų yra gana ryški. Matomas didelis atstumas tarp priešingų pažiūrų (pavyzdžiui, TSF (2) ir DPF (3)), pozicijos ir opozicijos (LSDPF (1) ir TSF (2)).

Nors vaizduose DS, gautuose naudojantis Euklido ir miesto kvartalo atstumais, grupės yra gana panašios, vienas LSDPF (1) narys vaizde, gautame naudojantis Euklido atstumais, yra TTLDF (5) klasteryje, ko nematyti vaizde, gautame naudojantis miesto kvartalo atstumais.

5.1.3. Bendrojo lavinimo mokyklos

Pastaraisiais metais sparčiai vykdant mokyklos reformą, tenka analizuoti ir pažinti giluminius procesus, vykstančius Lietuvos bendrojo lavinimo mokyklose. Tam reikia atskleisti mokytojų kvalifikacijos, amžiaus kaitą ir jų skaičiaus dinamiką. Taip pat reikia išryškinti tendencijas, vykstančias pirmaujančiose mokyklose ir paanalizuoti priežastis, stabdančias harmoningą mokyklų vystymąsi. Saviorganizuojantis neuroninis tinklas (SOM) ir Sammono projekcija buvo taikyti bendrojo lavinimo mokykloms palyginti [42].

Tyrimo pagrindas – Lietuvos Švietimo ir mokslo ministerijos (ŠMM) bei Atviros Lietuvos Fondo bendras projektas „Lietuvos pedagogų duomenų bazės kokybinis laidavimas, tyrimas (regioninė specifika) ir pedagogų poreikio prognozė“ (www.mii.lt/pedagogai/). Detalūs tyrimų rezultatai pateikti [40] monografijoje.

Tyrimo tikslas – sukurti metodą bendrojo lavinimo mokykloms tarpusavyje palyginti pedagoginio personalo kvalifikacijos, amžiaus ir kaitos požiūriu.

Duomenų šaltinis – Švietimo informacinių technologijų centre (ITC) nuo 1991 metų kaupiami duomenys apie pedagogus, dirbančius Lietuvos švietimo įstaigose: ikimokyklinėse įstaigose, pradinėse, pagrindinėse, vidurinėse mokyklose, gimnazijose, profesinėse, aukštesniosiose mokyklose, technikumuose, neformaliojo ir papildomojo ugdymo įstaigose. Iš šių pedagogų duomenų bazių analizei pasirinkta 19 Panevėžio miesto (tyrime jos pažymėtos numeriais nuo 1 iki 19) ir 9 Panevėžio rajono (pažymėtos nuo 20 iki 28) mokyklų. Nagrinėtos dvi gimnazijos (jų numeriai 1 ir 2), likusios – vidurinės mokyklos. Analizuojami mokytojai, dirbantys V–XII klasėse, taip pat gimnazinėse klasėse. Remtasi dviem grupėmis duomenų: 1997–1998 ir 1999–2000 m. m. Tokia analizė, įvertinant laiko faktorių, leidžia stebėti mokyklų panašumų ar skirtingumų kaitą kelių metų laikotarpiu, ieškoti tokios kaitos priežasčių. Analizės rezultatai pateikiami paprasta grafine forma, todėl juos galima lengvai interpretuoti.

Mokykloms apibūdinti yra pasirinkti rodikliai, kurie naudingi ir gilesnėms edukologinėms interpretacijoms:

x_1 – mokytojų ekspertų ir mokytojų metodininkų procentas,

x_2 – mokytojų, neturinčių tinkamos kvalifikacijos (dirbančių ne pagal savo turimą specialybę), procentas,

x_3 – mokytojų, kurių amžius viršija 55 metus, procentas,

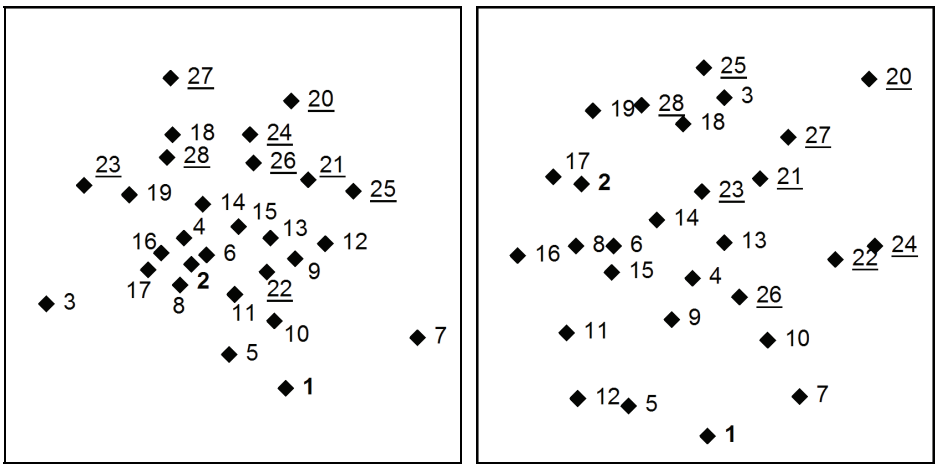
x_4 – mokytojų, kurių amžius iki 35 metų, procentas,

x_5 – mokytojų skaičiaus augimo (mažėjimo) procentas.

Rodikliai mokyklas apibūdina trimis požiūriais: x_1 ir x_2 nusako mokytojų kvalifikaciją, x_3 ir x_4 – amžių, x_5 – mokytojų skaičiaus dinamiką. Norėta pažiūrėti, kokią įtaką mokyklai daro šie rodikliai (parametrai), kaip mokyklos grupuojasi pagal pasirinktus parametrus ar grupės keičiasi laike. Kiekvienai mokyklai $X_i, i = 1,\ldots,m,$ iš pedagogų duomenų bazių apskaičiuojamos n-mačio (šiuo atveju $n = 5$) vektoriaus $X_i = (x_{i1}, x_{i2}, x_{i3}, x_{i4}, x_{i5})$ reikšmės. Rodikliai x_1, x_2, x_3, x_4, x_5 apibūdina m mokyklų įvairiais aspektais. Jų

matavimo skalės yra skirtingos, todėl prieš analizuojant duomenis būtina suvienodinti rodiklių mastelius. Duomenys normuoti taip, kad parametrų vidurkiai būtų lygūs 0, o dispersijos lygios 1 (žr. 1.3 skyrelyje).

Pirmiausia daugiamačiai duomenys vizualizuojami naudojantis Sammono algoritmu. Galima stebėti, kaip vektoriai $X_i = (x_{i1}, x_{i2}, x_{i3}, x_{i4}, x_{i5})$, $i = 1,\ldots,28$, apibūdinantys atskiras mokyklas, išsidėsto plokštumoje proporcingai jų tarpusavio panašumui. Tai pavaizduota 5.5 paveiksle. Skaičiai nurodo analizuojamų vektorių $X_1, X_2, \ldots, X_{28}$ numerius (indeksus). Būtina turėti omeny, kad numeriais 1 ir 2 pažymėtos gimnazijos (paveiksluose šie numeriai paryškinti), 3–19 – kitos Panevėžio miesto vidurinės mokyklos, o 20–28 – rajono mokyklos (paveiksluose šie numeriai pabraukti).



a) b)

5.5 pav. Panevėžio miesto ir rajono mokyklų išsidėstymas, gautas naudojantis Sammono algoritmu: a) 1997–1998 m. m., b) 1999–2000 m. m.

Iš 5.5 paveikslo dar pakankamai sunku vizualiai vertinti mokyklų panašumą, nes mokyklas atitinkantys taškai gana tolygiai pasiskirstę plokštumoje, nesimato jokių grupių. Tačiau, kaip matysime toliau, taikant saviorganizuojantį neuroninį tinklą (SOM), galima geriau pastebėti panašių mokyklų grupes. Analizuojant duomenis, naudotasi dviem SOM tinklų sistemomis (*SOM-PAK* ir *Kleiwego*, detaliau apie jas skaitykite 4.2.4 skyrelyje), taip pat SOM tinklo ir Sammono algoritmo nuosekliuoju junginiu (žr. 4.3.1 skyrelį).

Duomenų išdėstymas SOM tinkle parodytas 5.3 ir 5.4 lentelėse, taip pat 5.6 ir 5.7 paveiksluose. Visais atvejais – tai lentelės, kurių langeliai užpildyti mokyklų numeriais. Artimesniuose langeliuose „esančios“ mokyklos yra panašesnės, o tolimesniuose – labiau skirtingos. Paveiksluose panaudotos papildomos SOM tinklo vizualizavimo galimybės. Aiškiau 5.3 ir 5.4 lentelių duomenis galima suprasti kartu nagrinėjant ir 5.8 paveikslą, kur po SOM mokymo gauti vektoriai nugalėtojai yra analizuoti Sammono algoritmu. Reikia pastebėti, kad *SOM-PAK* sistemos rezultatai yra sunkiausiai suvokiami. Kiek lengviau yra nagrinėti *Kleiwego* sistemos rezultatus. Paprasčiausia interpretuoti SOM tinklo ir Sammono algoritmo junginio rezultatus, nes vizualiai matome SOM tinklu klasterizuotų taškų išsidėstymą plokštumoje.

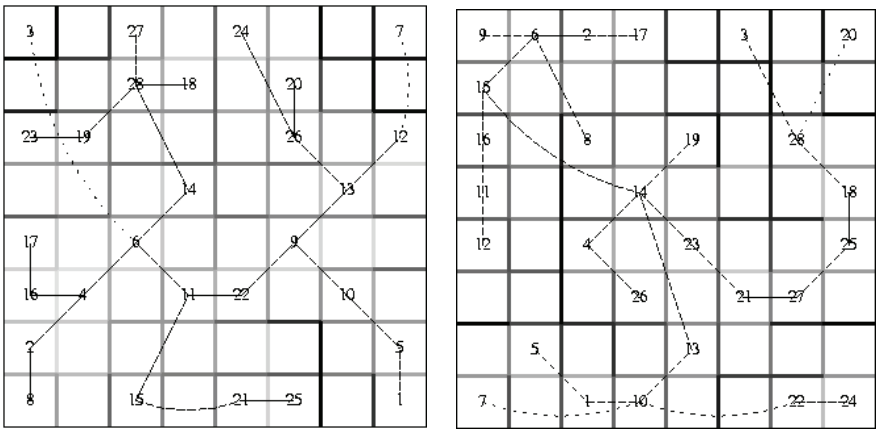
Pastebėsime (žr. 5.8 pav.), kad Panevėžio miesto ir rajono mokyklos sudaro kelias grupes. 1997–1998 m. m. išryškėja miesto mokyklų grupė, į kurią įeina viena rajono mokykla. Tai Pajstrio Juozo Zikaro vidurinė mokykla (23). Miesto mokyklos tarpusavyje sudaro dvi stambias grupes. Dalis grupuojasi apie Juozo Balčikonio gimnaziją (1), kitos – apie Vytauto Žemkalnio gimnaziją (2). Į rajono mokyklų grupę patenka Panevėžio miesto 3-ioji vidurinė mokykla (3), tai matyti ir 5.6a bei 5.7a paveiksluose. 1999–2000 m. m. taip pat matome miesto ir rajono mokyklų grupes (žr. 5.8b pav.), tik čia į miesto mokyklų grupę patenka Panevėžio rajono Smilgių vidurinė mokykla (26). Juozo Balčikonio gimnazijos (1) ryšys su kitomis miesto mokyklomis ima silpnėti. Tai matyti ir palyginus 5.3 ir 5.4 lenteles: 5.4 lentelėje ši mokykla yra atskirame langelyje.

5.3 lentelė. Panevėžio miesto ir rajono mokyklų išsidėstymas SOM [4x4] tinkle (1997–1998 m. m.)

3, 27	20, 24, 26		21, 25
18, 28			9, 13, 22
19, 23	14		10, 12
	2, 4, 8, 15, 16, 17	6, 11	**1**, 5, 7

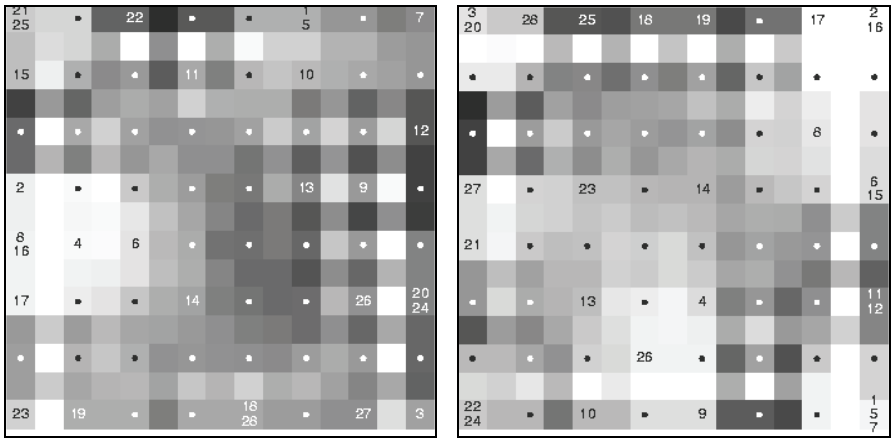
5.4 lentelė. Panevėžio miesto ir rajono mokyklų išsidėstymas SOM [4x4] tinkle, (1999–2000 m. m.)

20	3, 28	18	21, 25, 27
22, 24	13	23	19
7, 10	4, 9, 26		8, 14
1	5, 12	6, 11, 15	**2**, 16, 17



a) b)

5.6 pav. Panevėžio miesto ir rajono mokyklų išsidėstymas SOM [8x8] tinkle, gautame *Kleiwego* sistema: a) 1997–1998 m. m., b) 1999–2000 m. m.



a) b)

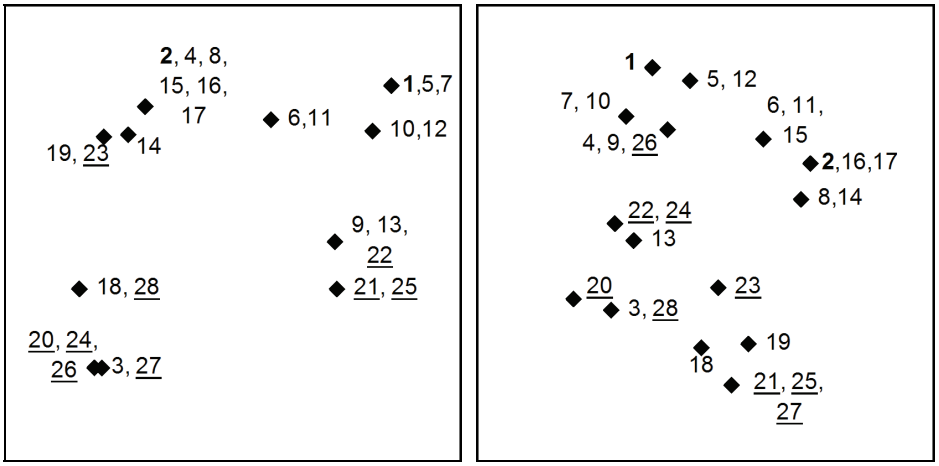
5.7 pav. Panevėžio miesto ir rajono mokyklų išsidėstymas SOM [8x8] tinkle, gautame *SOM-PAK* sistema: a) 1997–1998 m. m., b) 1999–2000 m. m.

Panevėžio miesto ir rajono pagrindu atliktas tyrimas parodė, kad mokyklas galima sėkmingai vertinti „miestas–kaimas“ ar „gimnazija–vidurinė mokykla“ aspektu. Jei mokykla būtų apibūdinama vienu ar dviem rodikliais, tai tokios vertinimo problemos nebūtų. Tačiau kai rodiklių turime daugiau – problema tampa sudėtinga. Turint tokios analizės rezultatus, tikslinga spręsti atvirkštinį

uždavinį – žinant mokyklos vietą kitų mokyklų kontekste, įvertinti, kokio rodiklių reikšmių rinkinio turėtų „siekti“ gera mokykla.

Grupę mokyklų, apibūdinamų pasirinkta rodiklių sistema, išanalizavus naudojantis SOM tinklu, Sammono algoritmu ir jų junginiu, galima vizualiai įvertinti mokyklų panašumus ir skirtingumus. Tokia kompleksinė analizė leidžia visapusiškai įvertinti sukauptus duomenis ir gauti vertingų žinių, kurios būtų naudingos optimizuojant bendrojo lavinimo mokyklų tinklą ir jų vadybą.

Pasiūlyta mokyklų analizės metodika yra universali – ji nepriklauso nuo pasirinktos rodiklių sistemos. Tai atveria plačias galimybes tokią sistemą tobulinti.



a) b)

5.8 pav. Panevėžio miesto ir rajono mokyklų išsidėstymas, gautas naudojantis SOM tinklo ir Sammono algoritmo junginiu: a) 1997–1998 m. m., b) 1999–2000 m. m.

5.2. Taikymai medicinoje ir farmakologijoje

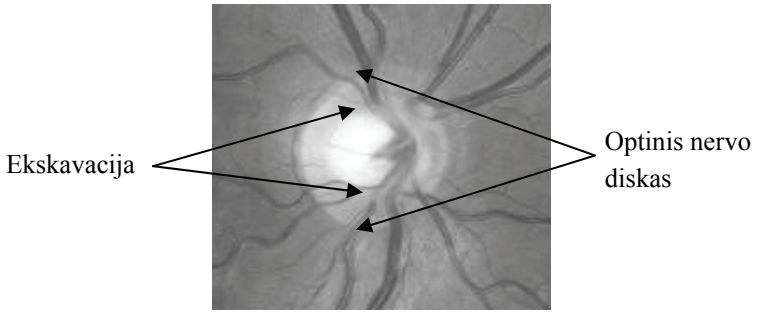
Šiame skyrelyje supažindinama su vizualiosios analizės rezultatais ir jų interpretacijomis tiriant įvairaus pobūdžio medicininius bei farmakologinius duomenis:

- oftalmologinius [11], [43];
- fiziologinius [9], [11];
- širdies ritmo;
- molekulinio sąryšio [159].

5.2.1. Oftalmologiniai duomenys

Analizuojami objektai – akies dugno parametrizuotos nuotraukos. Tokios nuotraukos pavyzdys pateiktas 5.9 paveiksle.

Yra išmatuoti akies dugno nuotraukos 27 skaitiniai parametrai, kurie gali apibūdinti tam tikras akių ligas. Pagrindinis šios analizės tikslas – stebėti, kaip vektoriai, sudaryti iš akių dugno nuotraukų skaitinių parametrų, išsidėsto plokštumoje, ar jie sudaro būdingas grupes, ar sveikas akis atitinkantys taškai atsiskiria nuo pažeistas akis atitinkančių taškų [11], [43].



5.9 pav. Akies dugno nuotrauka

Matuoti kelių grupių parametrai.

a) akies dugno nervo optinio disko (OND) parametrai:

x_1 – OND aproksimuojančios elipsės ilgoji ašis,
x_2 – trumpoji ašis,
x_3 – ilgasis spindulys,
x_4 – trumpasis spindulys,
x_5 – horizontalusis diametras,
x_6 – vertikalusis diametras,
x_7 – plotas,
x_8 – ekscentricitetas,
x_9 – perimetras;

b) ekskavacijos (EKS) parametrai (*ekskavacija* – tai įdubimas, esantis optinio nervo disko centre, 5.9 paveikslo nuotraukoje tai pašviesėjusi optinio nervo disko dalis):

x_{10} – EKS aproksimuojančios elipsės ilgoji ašis,
x_{11} – trumpoji ašis,
x_{12} – ilgasis spindulys,
x_{13} – trumpasis spindulys,
x_{14} – horizontalusis diametras,
x_{15} – vertikalusis diametras,
x_{16} – plotas,
x_{17} – ekscentricitetas,
x_{18} – perimetras;

c) santykiai tarp įvairių OND, EKS, NRK parametrų (*neuroretinalinis kraštas* (NRK) – tai audinys, esantis tarp išorinio ekskavacijos krašto ir išorinio optinio nervo disko krašto):

x_{19} – EKS ir OND horizontaliųjų diametrų santykis,
x_{20} – EKS ir OND vertikaliųjų diametrų santykis,
x_{21} – NRK plotas,
x_{22} – NRK ir OND plotų santykis,
x_{23} – EKS ir OND plotų santykis;

d) neuroretinalinio krašto (NRK) dalių storiai:

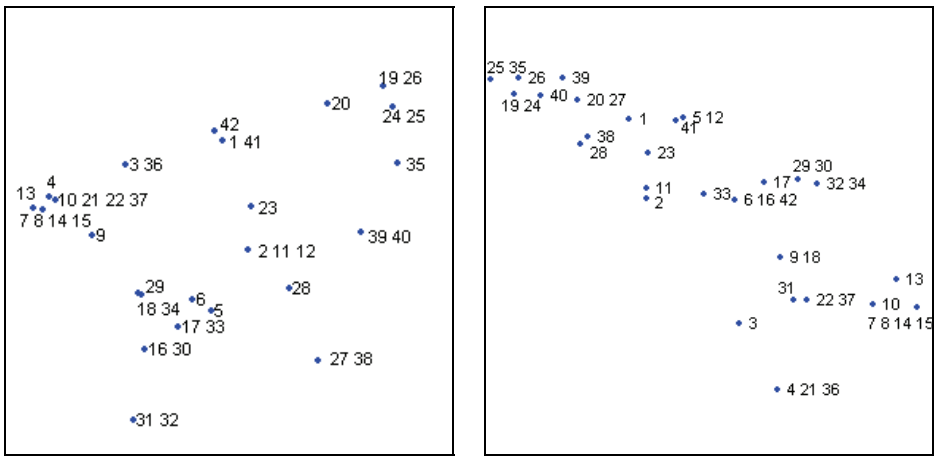
x_{24} – apatinės srities storis,
x_{25} – viršutinės srities storis,
x_{26} – nosies srities storis,
x_{27} – smilkininės srities storis.

Parametrų $x_1, x_2, \ldots, x_{27}$ reikšmės buvo apskaičiuotos ar išmatuotos 42 pacientams. Sudaryti pacientus apibūdinantys vektoriai $X_1, X_2, \ldots, X_{42}$, ($X_i = (x_{i1}, x_{i2}, \ldots, x_{i27})$, $i = 1, \ldots, 42$). *Glaukoma* diagnozuota 18 pacientų, t. y. vektoriai $X_1, X_2, \ldots, X_{18}$ apibūdina glaukomą (jų numeriai – 1–18). *Glaukoma* – tai optinio nervo disko ligų grupė. Šios ligos atsiranda dėl tinklainės nervinių ląstelių sumažėjimo ir joms būdingas ekskavacijos padidėjimas. Likusiems 24 pacientams glaukoma nediagnozuota. Juos apibūdina vektoriai $X_{19}, X_{20}, \ldots, X_{42}$ (jų numeriai – 19–42). Duomenys apie šių pacientų akis gali būti naudojami kaip nusakantys sveikas akis tiriant pacientų, kuriems diagnozuota glaukoma, duomenis.

Iš pradžių SOM tinklo ir Sammono algoritmo junginiu buvo analizuojami vektoriai, sudaryti iš visų 27 parametrų $x_1, x_2, \ldots, x_{27}$. Iš gauto 5.10a paveikslo gana sudėtinga atskirti sveikas akis nuo ligotų. Sveikas akis nusakantys vektoriai lyg ir bando sudaryti vieną grupę (X_4, X_7, X_8, X_9, X_{10}, X_{13}, X_{14}, X_{15}), tačiau į ją taip pat įsimaišo ir atitinkantys glaukomos pažeistas akis (X_{21}, X_{22}, X_{37}). Galima to priežastis – didelis parametrų skaičius. Gal neverta imti visų 27 parametrų, nes juos spaudžiant į dvimatę erdvę neišvengiami dideli iškraipymai.

Analizuojant parametrų individualias grupes, galima pastebėti tam tikrus reguliarumus. Vektorių, sudarytų tik iš 7 ekskavacijos parametrų (x_{10}, x_{11}, $x_{14} - x_{18}$), projekcijos plokštumoje pavaizduotos 5.10b paveiksle. Čia galima pastebėti, kad dauguma taškų, kuriuos atitinka sveikas akis apibūdinantys vektoriai (X_7, X_8, X_{10}, X_{13}, X_{14}, X_{15}), išsidėsto viename kampe, o priešingame kampe – tik taškai, atitinkantys pažeistas glaukomos akis nusakančius vektorius.

Peržiūrint taškus nuo viršutinio kairiojo 5.10b paveikslo kampo ir slenkant į dešinįjį apatinį šio paveikslo kampą, galima pastebėti, kad iš pradžių yra išsidėstę taškai, atitinkantys pažeistas akis aprašančius vektorius, vėliau jų mažėja ir galiausiai lieka tik taškai, kurie atitinka sveikas akis apibūdinančius vektorius.



a) b)

5.10 pav. Vektorių, atitinkančių oftalmologinius duomenis, projekcijos plokštumoje, analizuojant: a) visus 27 parametrus, b) tik 7 ekskavacijos parametrus

Oftalmologinių parametrų sistemos tyrimas

Kaip matyti iš pateiktų tyrimų rezultatų, formuojant vektorius iš visų 27 parametrų ir juos spaudžiant į dvimatę erdvę, atsiranda dideli iškraipymai. Tai gali būti dėl didelio parametrų skaičiaus. Vienas iš būdų atrinkti pagrindinius parametrus – analizuoti jų koreliacines matricas (apie koreliacinių matricų vizualiosios analizės teorinį pagrindą skaitykite 5.3.1 skyrelyje).

Duomenys analizei buvo paruošti tokiu būdu: paimta visa duomenų aibė (visas ligų pažeistas ir sveikas akis atitinkantys vektoriai – taigi iš viso 138 27-mačiai vektoriai); apskaičiuota parametrų $x_1, x_2, \ldots, x_{27}$ koreliacinė matrica; iš jos 5.3.1 skyrelyje aprašytu metodu atkurta vektorių sistema, gauti 27 27-mačiai vektoriai. Vienas analizuojamas vektorius atitinka vieną iš parametrų $x_1, x_2, \ldots, x_{27}$. Tikslas – stebėti, kaip parametrai išsidėsto plokštumoje, kokios jų grupės susidaro. Iš pradžių analizuojamais vektoriais mokomas SOM tinklas. Jo mokymo rezultatai gali priklausyti nuo kelių sąlygų: tinklo neuronų pradinių reikšmių parinkimo, analizuojamų vektorių pateikimo į tinklą eilės, tinklo mokyme atliktų epochų skaičiaus. Todėl tikslinga kelis kartus mokyti SOM tinklą, parinkus įvairias pradines sąlygas, ir išrinkti tą, kuriuo gaunama mažiausia SOM kvantavimo paklaida $E_{SOM(q)}$, apibrėžiama (4.14) formule. Radus geriausias kvantavimo paklaidos prasme sąlygas ($E_{SOM(q)} = 0{,}257$), taikomas SOM tinklo ir Sammono projekcijos integruotojo junginio algoritmas, randamas daugiamačių vektorių išdėstymas plokštumoje (žr. 5.11a pav.).

Iš 5.5 lentelės matyti, kaip parametrai $x_1, x_2, \ldots, x_{27}$ išsidėsto SOM tinkle. Atlikta 500 SOM mokymo epochų, SOM mokymo procesas buvo padalytas į 50 blokų, Sammono algoritmo iteracijų skaičius lygus 500, gauta Sammono projekcijos paklaida $E_S = 0{,}034$ (I eksperimentas). Ir iš 5.11a paveikslo, ir iš 5.5 lentelės matyti kelios susidariusios parametrų grupės, čia nurodyti skaičiai yra parametrų numeriai. Kalbant apie stambesnes grupes (jos paveiksluose apibrėžtos) galima išskirti tokias: $\{x_1 - x_7, x_9\}$ – tai OND elipsės parametrai, $\{x_{10} - x_{16}, x_{18}\}$ – tai EKS elipsės parametrai, likusieji parametrai $\{x_{19} - x_{27}\}$ taip pat bando sudaryti grupę. Tačiau parametrai x_8 ir x_{17} (tai OND ir EKS elipsių ekscentricitetai) nutolę nuo kitų grupių.

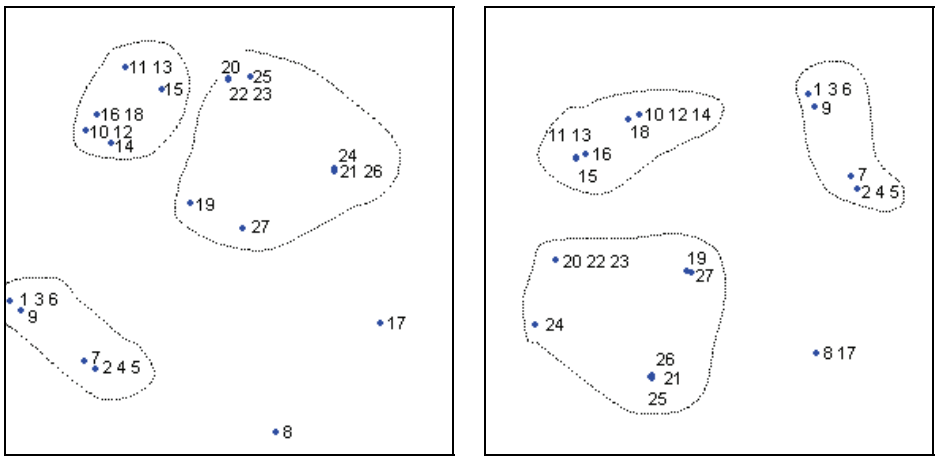
Kiekvienoje stambesnėje grupėje galima įžvelgti kelias smulkesnes grupeles, pavyzdžiui, OND elipsės parametrai x_1, x_3, x_6, x_9 šiek tiek atsiskiria nuo parametrų x_2, x_4, x_5, x_7. Tą patį galima pasakyti ir apie kitas grupes. Panašios grupės susidaro ir SOM tinkle (žr. 5.5 lentelę).

5.5 lentelė. Oftalmologinių parametrų išsidėstymas SOM [6x6] tinkle (I eksperimentas)

11, 13		16, 18	10, 12		2, 4, 5
	15		14		7
20	22, 23		19		9
25			27		1, 3, 6
21, 26	24		17		8

5.6 lentelė. Oftalmologinių parametrų išsidėstymas SOM [6x6] tinkle (II eksperimentas)

1, 3, 6		10, 12, 14	18	16	11, 13
9					15
7		27	19		
2, 4, 5					20, 22, 23
		21	25		
8, 17		26			24



a) b)

5.11 pav. Oftalmologinių parametrų išsidėstymas plokštumoje, gautas naudojantis SOM [6x6] tinklo ir Sammono algoritmo junginiu:
a) I eksperimentas, b) II eksperimentas

Taip pat buvo atlikti eksperimentai ir esant kitoms SOM tinklo mokymo sąlygoms (II eksperimentas). Pastebėta, kad stambiosios parametrų grupės visada išlieka tos pačios. Tuo tarpu kai kurios smulkesnės grupelės truputėlį persigrupuoja. Tai galima įsitikinti 5.11b paveikslą ir 5.6 lentelę palyginus su 5.11a paveikslu ir 5.5 lentele. Čia 5.11b paveikslas ir 5.6 lentelė gauti atlikus 300 SOM tinklo mokymo epochų. SOM tinklo mokymo procesas buvo padalytas į 30 blokų. Sammono algoritmo iteracijų skaičius lygus 1000. Gauta kvantavimo paklaida $E_{SOM(q)} = 0{,}297$, Sammono projekcijos paklaida $E_S = 0{,}038$. Iš čia matome, kad parametrai x_8 ir x_{17} yra nutolę nuo savo grupių, tačiau yra arti vienas kito ir sudaro atskirą grupelę.

Pagal gautų paklaidų reikšmes I eksperimento rezultatai yra kiek geresni, tačiau ir II eksperimentas yra naudingas parametrų sąryšiams vertinti, ypač kai stebime abiejų eksperimentų rezultatus kartu.

Iš gautų rezultatų galima būtų padaryti tokias bendras išvadas:

- Optinį nervo diską (OND) aproksimuojančios elipsės parametrai $x_1 - x_7, x_9$ sudaro atskirą stambią grupę, kuri yra šiek tiek toliau nuo kitų parametrų. Taip ir turėtų būti, nes OND elipsės parametrai visai neapibūdina ligos. Šioje parametrų grupėje galima įžvelgti dvi mažesnes grupeles $\{x_1, x_3, x_6, x_9\}$ ir $\{x_2, x_4, x_5, x_7\}$. Tai matyti abejuose 5.11 paveiksluose.
- Ekskavaciją (EKS) aproksimuojančios elipsės parametrai $x_{10} - x_{16}$, x_{18} taip pat sudaro atskirą grupę. Joje susidaro dvi mažesnės grupelės $\{x_{11}, x_{13}, x_{15}\}$ ir $\{x_{10}, x_{12}, x_{14}, x_{18}\}$, parametras x_{16} kartais yra arčiau pirmos grupelės (žr. 5.11b pav.), kartais – arčiau antrosios (žr. 5.11a pav.).
- Parametrai x_8 ir x_{17} (OND ir EKS elipsių ekscentricitetai) nutolę nuo kitų tų elipsių parametrų. Vienu atveju jie yra nutolę ir vienas nuo kitos (5.11a pav.), kitu jie sudaro atskirą grupę (5.11b pav.).
- Dar viena grupę sudaro parametrai $x_{19} - x_{27}$. Tai santykiai tarp OND ir EKS elipsių parametrų, ir neuroretinalinio krašto (NRK) parametrai. Tačiau grupės viduje įžvelgiamos kelios smulkesnės grupelės. Pastebėsime, kad yra nusistovėjusios grupės $\{x_{20}, x_{22}, x_{23}\}$ ir $\{x_{21}, x_{26}\}$, o ryšiai tarp kitų parametrų nėra tokie stiprūs, pavyzdžiui, vienu atveju x_{19} ir x_{27} yra toliau vienas nuo kito, kitu atveju labai arti.

5.2.2. Fiziologiniai duomenys

Fiziologinių duomenų analizės tikslas – įvertinti žmogaus sveikatos būklę ir jo galimybes sportuoti. Tai ypač aktualu kineziologams ir sporto medikams. Analizuojama aibė sudaryta iš duomenų trijų grupių vyrų: sergančių išemine širdies liga (I grupė, 61 tiriamasis), sveikų, nesportuojančių (II grupė, 110 tiriamųjų) ir profesionalių sportininkų (III grupė, 161 tiriamasis) [9], [11].

Tyrimui buvo pasirinkti paprasčiausiai ir lengviausiai nustatomi elektrokardiogramos (EKG) bei arterinio kraujo spaudimo (AKS) dydžiai:

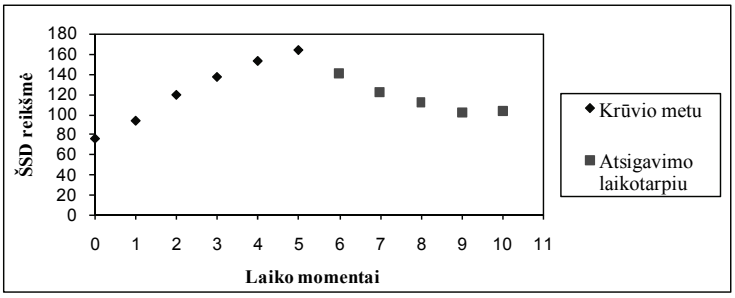
- širdies susitraukimų dažnis (ŠSD);
- sistolinis arterinio kraujo spaudimas (S);
- diastolinis arterinio kraujo spaudimas (D);
- intervalas elektrokardiogramoje nuo jungties taško J iki T bangos pabaigos (JT intervalas).

Šie keturi skaitiniai dydžiai yra fiksuojami tam tikrais laiko momentais, atliekant ergometrinio dviračio tyrimą. Dar papildomai buvo apskaičiuoti du išvestiniai dydžiai: (S-D)/S ir JT/RR (RR = 60/ŠSD).

Prieš pradedant tyrimą, tiriamiesiems išmatuojamosi keturios fiziologinės charakteristikos: ŠSD, S, D, JT. Pradinė galia – 50W. Tyrimo metu kas minutę galia didinama po 50W. Prieš tai matuojamos nurodytos keturios charakteristikos. Tyrimas baigiamas, kai tiriamasis nebepajėgia daugiau jo atlikti arba gydytojas pastebi esminius pakitimus širdies veikloje ir liepia tyrimą baigti. Po to seka organizmo atsigavimo laikotarpis, kurio metu kas minutę vėl matuojami tie patys keturi dydžiai.

Vieno tyrimo metu gaunamas kelių dydžių reikšmių rinkinys, be to, kiekvieno tiriamojo fiziologinių charakteristikų reikšmių skaičius skiriasi, nes jis priklauso nuo maksimalios galios, kuriai esant tiriamasis pajėgė įveikti fizinį krūvį. Vieno tiriamojo širdies susitraukimų dažnio (ŠSD) reikšmės, matuotos tam tikrais laiko momentais didinant krūvį ir atsigavimo laikotarpiu, pavaizduotos 5.12 paveiksle.

Toks reikšmių kitimas atspindi žmogaus širdies veiklą. Visos keturios fiziologinės charakteristikos (ŠSD, JT, S, D) yra svarbios, todėl būtina analizuoti jų visumą. Tokia integrali analizė gydytojui – gana sudėtingas uždavinys. Norint jį supaprastinti, būtina rasti šios dinamikos integralius įverčius, kurie būtų lengviau suvokiami. Tam gali būti naudojamos įvairios strategijos. Viena iš jų paremta fraktalinių dimensijų skaičiavimais. Ja naudotasi [47] darbe.



5.12 pav. Širdies susitraukimų dažnio kitimas krūvio metu ir atsigavimo laikotarpiu

Fraktalinės dimensijos skaičiuojamos tokiu būdu:

1. Pradiniai duomenys – tai tyrimo metu gautos visų nagrinėjamų dydžių skaitinės reikšmės, esant tam tikroms galioms krūvio metu, taip pat penkios reikšmės atsigavimo po krūvio laikotarpiu. Gauti taškiniai įverčiai yra interpoliuojami kubiniu splainu. Tokiu būdu gaunamas kiekvieno dydžio funkcijos grafikas.
2. Fraktalinės dimensijos (užimtumo, informacinė, koreliacijos) skaičiuojamos remiantis gautais grafikais pagal tokią schemą:
   - nagrinėjamas kvadratinis tinklelis (langelio didumas lygus ε), uždėtas ant stebimo dydžio funkcijos grafiko;
   - kiekvienoje tinklelio dalyje suskaičiuojamas į ją patekusių taškų skaičius n_i ir padalijamas iš bendro taškų skaičiaus $\overline{n}$, t. y.

$$p_i(\varepsilon) = \frac{n_i}{\overline{n}};$$

   - apibrėžiama informacinė funkcija

$$I = -\sum_{i=1}^{n_\varepsilon} p_i(\varepsilon) \log[p_i(\varepsilon)],$$

   čia n_ε – užimtų langelių skaičius. Tuomet informacinė dimensija apbrėžiama taip:

$$d_{\inf} = -\lim_{\varepsilon \to 0} \frac{I}{\log(\varepsilon)} = \lim_{\varepsilon \to 0} \sum_{i=1}^{n_\varepsilon} \frac{p_i(\varepsilon) \log\left[p_i(\varepsilon)\right]}{\log(\varepsilon)}.$$

   Užimtumo dimensija gaunama pakeitus $I = \log n_\varepsilon$, o koreliacijos dimensija, kai $I = \log \sum_i \left(p_i(\varepsilon)\right)^2$.

Užimtumo, informacinė ir koreliacijos dimensijos apskaičiuotos keturioms matuotoms charakteristikoms (ŠSD, S, D, JT) ir dviem išvestinėms ((S-D)/S ir JT/RR (RR = 60/ŠSD)). Nustatyta, kad informatyviausia yra informacinė dimensija, todėl analizuojami daugiamačiai vektoriai $X_1, X_2, ..., X_{332}$ sudaryti būtent iš informacinės dimensijos 6 parametrų $x_1, x_2, ..., x_6$.

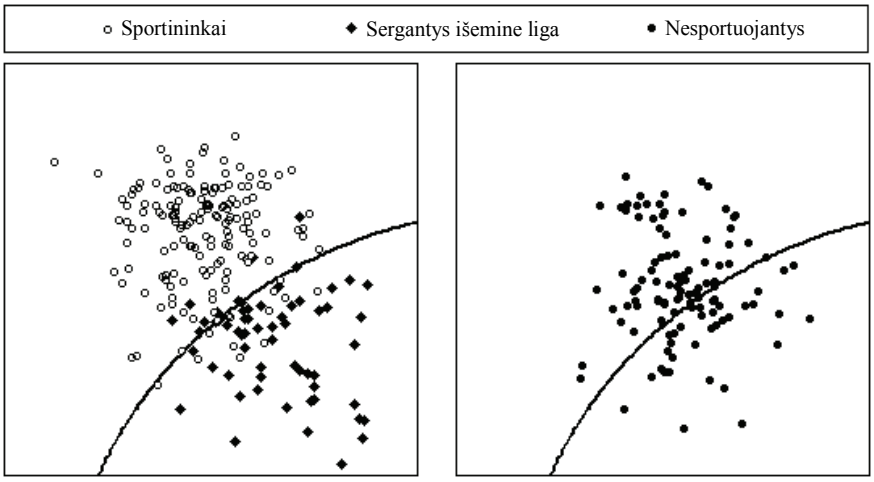
Iš pradžių SAMANN tinklas buvo mokomas dviejų grupių (I ir III) duomenimis. Naudoti šie tinklo ir jo mokymo parametrai:

- vienas paslėptasis neuronų sluoksnis, paslėptųjų neuronų skaičius $n_l = 20$;
- išėjimų skaičius $d = 2$;
- mokymosi parametras $\eta = 1$;
- momento konstantos reikšmė $\alpha = 0,3$;
- mokymo iteracijų skaičius yra 10000.

Mokymo metu surasti SAMANN tinklo išėjimai dviem duomenų aibės grupėms (I ir III) ir apskaičiuotos neuroninio tinklo svorių reikšmės. SAMANN tinklo išėjimai – tai analizuojamų šešiamačių duomenų projekcijos plokštumoje, t. y. dvimačiai vektoriai.

Analizuojamų grupių taškams atskirti naudotasi atraminių vektorių klasifikatoriumi (*support vector machine*, SVM) [9], [46]. Naudojant šį klasifikatorių, klasifikuoti ne daugiamačiai vektoriai, o jų projekcijos plokštumoje, gautos algoritmu SAMANN. Tyrimai [9] parodė, kad klasifikavimas, tiesiogiai naudojant daugiamačius duomenis, o ne jų dvimates projekcijas, yra panašios kokybės, kaip ir dvimačių projekcijų analizės, tik daugiamatėje erdvėje neturima galimybės betarpiškai vizualiai stebėti taškų padėtį klasių skiriamojo paviršiaus atžvilgiu.

Dviejų grupių (I ir III) projekcijos plokštumoje ir atraminių vektorių klasifikatoriumi gautas skiriamasis paviršius, atskiriantis I grupę (sergantieji išemine širdies liga) nuo visų kitų taškų, pavaizduotas 5.13a paveiksle. Turint šių dviejų grupių (I ir III) projekcijas plokštumoje, galima greitai ir pakankamai tiksliai rasti naujos grupės (sveikų, nesportuojančiųjų) projekcijas plokštumoje, t. y. atvaizduoti tą grupę be papildomų skaičiavimų. Tam ši grupė (110 taškų) buvo pateikta SAMANN tinklui. Žinant neuroninio tinklo svorių reikšmes, rastos šios grupės taškų projekcijos be papildomo tinklo mokymo (žr. 5.13b pav.). „Uždėjus“ 5.13a ir 5.13b paveikslus vieną ant kito, atsižvelgiant į skiriamąjį paviršių, galima daryti preliminarias išvadas apie tiriamųjų sveikatos būklę.



5.13 pav. Fiziologinių duomenų vaizdas plokštumoje, gautas naudojantis algoritmu SAMANN : a) sergančiųjų išemine liga ir sportininkų grupės, b) tik nesportuojančiųjų grupė

5.2.3. Širdies ritmo parametrų analizė miego stadijoms nustatyti

Miego sutrikimams ir ligoms nustatyti svarbi paciento miego struktūra, kuri parodo miego stadijų kaitą. Šiuolaikiniai miego stadijų nustatymo ir registravimo būdai yra pagrįsti polisomnografija, kuri matuoja smegenų aktyvumą elektroencefalografija, akių judesius elektrookulografija, raumenų aktyvumą elektromiografija, širdies ritmą elektrokardiografija ir kvėpavimo funkciją. Tokie tyrimai atliekami specialioje miego laboratorijoje, pavyzdžiui, Kauno medicinos universiteto Psichofiziologijos ir reabilitacijos institute. Tokie būdai yra lėti ir brangūs, dėl to ieškoma naujų.

Vien širdies ritmo registravimas yra daug pigesnis ir paprastesnis, dėl to miego stadijų identifikavimas analizuojant širdies ritmo parametrus yra aktualus praktinės medicinos požiūriu. Tačiau toks būdas nėra pakankamai patikimas. Dažniausiai analizuojami tiesiniai širdies ritmo parametrai, gauti naudojantis spektrine analize.

Pastaruoju metu kyla susidomėjimas ir netiesinių metodų panaudojimu, tačiau atskirai naudojami tiesiniai ir netiesiniai širdies ritmo parametrai negarantuoja patikimų rezultatų. Šiame skyrelyje tiesiniai ir netiesiniai širdies ritmo parametrai vizualizuojami siekiant nustatyti informatyvų jų rinkinį.

Tyrimui pasirinktas pirmosios lėtojo miego stadijos ir greitojo miego (kuriam būdingi greiti akių judesiai, dėl to jis vadinamas REM – *rapid eye movement*) atskyrimas. Esant šioms miego stadijoms, širdies ritmo spektriniai parametrai yra panašūs. Dėl to jas atskirti analizuojant spektrinius parametrus yra sudėtinga.

Buvo analizuojami ritmogramų įrašai, surinkti Kauno medicinos universiteto Psichofiziologijos ir reabilitacijos institute (http://www.pri.kmu.lt/datbank). Šiuose įrašuose širdies ritmas užrašytas RR (elektrokardiogramos sistolinių pikų) intervalų (t. y. laiko tarpų tarp dviejų tokių pikų) seka. RR sekos yra sinchronizuotos su polisomnografiniu būdu nustatytų miego stadijų indikatoriais. Išrinkti REM ir pirmąją miego stadiją atitinkantys RR sekų segmentai, kurių trukmė yra ne mažesnė negu viena minutė: 267 REM miego ir 109 pirmosios miego stadijos segmentai. Įvertinti šių segmentų RR laiko eilučių devyni žemiau pateikti statistikiniai, spektriniai ir netiesinės dinamikos parametrai.

Statistikiniai parametrai:

- vidurkis (μ);
- standartinis nuokrypis (σ).

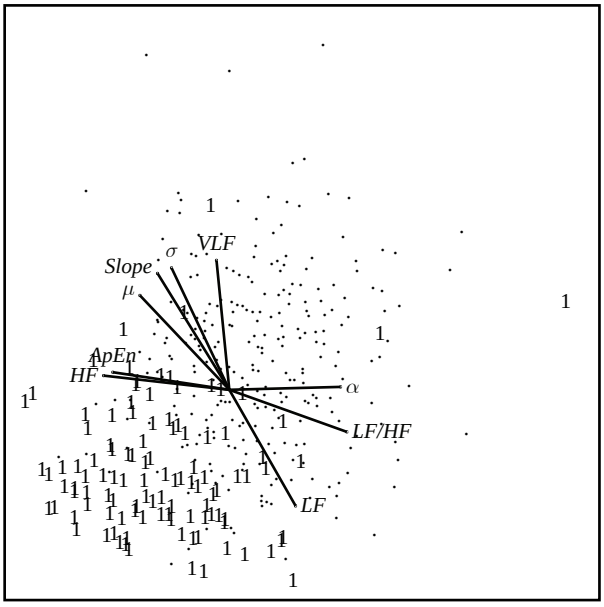
Įvertintas galios spektras buvo parametrizuotas, t. y. nustatyta galios dalis atitinkamiems dažnių intervalams. Iš jų suformuoti spektriniai parametrai:

- labai žemo dažnio komponentės 0,01–0,05 Hz (*VLF*) dalis;
- žemo dažnio komponentės 0,05–0,15 Hz (*LF*) dalis;
- aukšto dažnio komponentės 0,15–0,5 Hz (*HF*) dalis;
- žemo ir aukšto dažnių komponenčių dalių santykis (*LF/HF*).

Netiesinės dinamikos parametrai:

- aproksimacinė entropija (*ApEn*) [121];
- betrendės fliuktuacinės analizės (*detrended fluctuation analysis*, *DFA*) fraktalinio mastelio laipsnis (*fractal scaling exponent*) α [19];
- augimo statumas (*Slope*), nustatytas progresyvios betrendės fliuktuacinės analizės (*progressive detrended fluctuation analysis*) [136] metodu.

Daugiamačių duomenų masyvą sudaro 376 devynmačiai vektoriai, atitinkantys normuotas, kad būtų tarp nulio ir vieneto, devynių parametrų, apskaičiuotų RR sekų segmentams, reikšmes. Devynmačių duomenų vaizdas dvimatėje skalėje pateiktas 5.14 paveiksle.

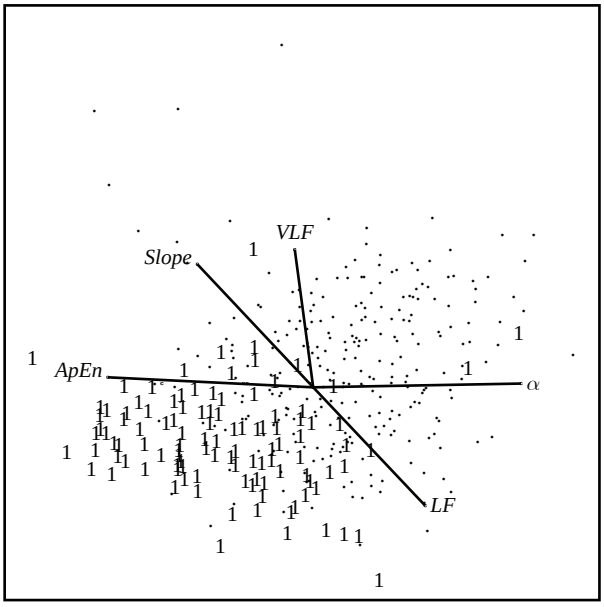


5.14 pav. REM ir pirmosios miego stadijos segmentų devynių RR parametrų vaizdas

Masyvą sudaro dviejų rūšių vektoriai: atitinkantys REM (.) ir pirmosios stadijos (1) miegą. Atkarpos rodo atitinkamų duomenų erdvės koordinatinių vektorių projekcijas į vaizdų plokštumą, t. y. jos jungia koordinačių pradžios duomenų erdvėje vaizdą su taškų, nutolusių nuo koordinačių pradžios viena iš koordinačių kryptimi duomenų erdvėje per vienetą, vaizdais. Šios atkarpos pažymėtos koordinačių ašis atitinkančių parametrų pavadinimais.

Paveiksle *HF* ir *ApEn* ašių projekcijos beveik pilnai sutampa. Trijų koordinatinių vektorių μ, σ ir *Slope* projekcijos gana artimos. Šie panašumai pagrindžia hipotezę apie parametrų skaičiaus sumažinimo galimybę, atmetant parametrus μ, σ ir *HF*. Taip pat atmestinas parametras *LF/HF*, kuris mažiau informatyvus negu *LF*.

Apibendrinami parametrų informatyvumo tyrimo rezultatus, REM ir pirmosios miego stadijos atskyrimui galime imti šiuos parametrus: *VLF*, *LF*, *ApEn*, α ir *Slope*. Įdomu palyginti originalių ir sumažintų duomenų aibių struktūrą, sugretinus devynmačių ir penkiamačių vektorių aibių dvimačius vaizdus. Sumažintų duomenų (palikus tik penkis parametrus) vaizdą matome 5.15 paveiksle. Iš tikrųjų, 5.14 ir 5.15 paveikslai labai panašūs tiek vaizdo taškų, tiek ir koordinatinių vektorių projekcijų tarpusavio išsidėstymu. Tai rodo, kad pasirinktų penkių parametrų pilnai pakanka šiam uždaviniui spręsti.



5.15 pav. REM ir pirmosios miego stadijos segmentų penkių RR parametrų vaizdas DS

5.2.4. Farmakologinis sąryšis

Baltymai yra sudėtingos organinės makromolekulės, sudarytos iš amino rūgščių, sujungtų peptidinėmis jungtimis. Baltymų yra ląstelėse ir virusuose. Baltymai reikalingi ląstelių ir audinių struktūroms formuoti. Prisijungdami mažesniąsias molekules (ligandus) jie vykdo signalines, transportavimo ir katalizavimo funkcijas. Konkretaus baltymo amino rūgščių seka, nusakyta jį koduojančio geno, lemia baltymo struktūrą ir funkciją.

Farmakologiniai duomenys – tai sulaikymo (*inhibition*) konstantų reikšmės, nusakančios konkretaus ligando ir priimančio baltymo sąryšį (*binding affinity*) tam tikromis eksperimento sąlygomis. Sulaikymo konstantos yra išmatuojamos konkuruojančio sąryšio bandymais, testuojant tam tikro ligando gebėjimą pakeisti radioaktyvų ligandą sąryšio vietoje. Tiriami ligandai gali būti natūralūs neurotransmiteriai ar vaistai, kurie, jungdamiesi prie baltymo, gali aktyvuoti arba blokuoti jo veiklą. Paprastai farmakologinio sąryšio duomenys pateikiami matrica, kurios vienas matmuo atitinka ligandus, tirtus eksperimentų metu, kitas – skirtingus baltymus, prie kurių pastarieji prisijungia. Baltymai gali būti skirtingų rūšių gyvūnų ar net sukurti mutuojant natūralius baltymus. Net ir labai skirtingų rūšių gyvūnų atitinkami baltymai

gali būti panašūs, dėl to galėtų būti naudojami vaistų tyrimams. Tačiau gali būti pakankamai skirtingi ir tos pačios rūšies gyvūno. Farmakologinių duomenų analizė yra labai svarbi. Struktūrinių grupės ligandų savybių ir sąryšio įvairiose rūšyse bei porūšiuose panašumas gali suteikti naudingos informacijos kuriant vaistus. Kita vertus, farmakologinių baltymų savybių analizė gali padėti nustatyti struktūrines jų savybes. Vizuali farmakologinių duomenų matricos analizė yra sudėtinga. Pagrindinių komponenčių analize ir dvinariais klasifikavimo medžiais naudotasi [126] darbe. Dvinariai medžiai klasterizuoja objektus poromis, leidžia euristiškai analizuoti ir interpretuoti duomenis, bet gali būti klaidinantys. Vizualizuojant pagrindines komponentes, dalis informacijos yra ignoruojama. Nors minėtame darbe buvo vizualizuojamos trys pagrindinės komponentės, 15–19% duomenų jose nebuvo atspindėta.

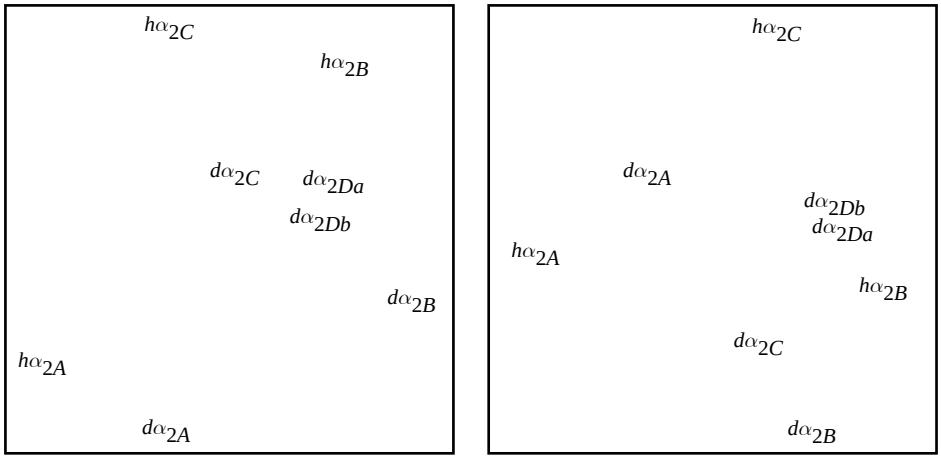
Šiame skyrelyje aprašoma, kaip DS su Euklido ir miesto kvartalo atstumais taikomos farmakologinio sąryšio duomenims vizualizuoti ir analizuoti. Baltymų skirtingumas yra įvertinamas atstumu tarp baltymus atitinkančių $\log_{10}$-transformuotų farmakologinių duomenų matricos vektorių. Panašiai ligandų skirtingumas yra įvertinamas atstumu tarp jas atitinkančių $\log_{10}$-transformuotų vektorių.

Vizualizuojami įvairūs farmakologinio sąryšio duomenys [159]:

- trijų žmogaus ir penkių žuvytės zebrinės danijos (*Danio rerio*) α_2-adrenoceptorinių baltymų ($m = 8$) ir 20 ligandų ($m = 20$), kurie jungiasi su tais baltymais, duomenys [126];
- žmogaus, žiurkės, jūrų kiaulytės ir kiaulės α_2-adrenoceptorinių baltymų ($m = 12$) duomenys [140];
- laukinių ir mutuotų α_1-adrenoceptorinių baltymų ($m = 12$) duomenys [76].

Pirmosios duomenų aibės 20 ligandų parametrų vaizdai buvo analizuoti 3.4 ir 3.5 skyreliuose. Šiame skyrelyje pateiksime ir aptarsime baltymų parametrų vaizdus DS.

Farmakologinių duomenų [126] trijų žmogaus ir penkių žuvytės zebrinės danijos α_2-adrenoceptorinių baltymų vaizdai DS pateikti 5.16 paveiksle. Žmogaus baltymai pažymėti pirma raide „*h*“, o žuvytės – „*d*“. Iš paveikslo matyti, kad $d\alpha_{2Da}$ ir $d\alpha_{2Db}$ yra glaudžiai susigrupavę. Gali būti įžvelgiamas žuvytės baltymų grupavimasis centre, o žmogaus baltymų – aplink. Arčiau vienas kito yra $h\alpha_{2B}$ ir $d\alpha_{2Da}$ vaizdai negu $h\alpha_{2A}$ ir $d\alpha_{2A}$, $h\alpha_{2B}$ ir $d\alpha_{2B}$, ar $h\alpha_{2C}$ ir $d\alpha_{2C}$. Tai ypač aiškiai matyti vaizde DS, gautame naudojantis miesto kvartalo atstumais.

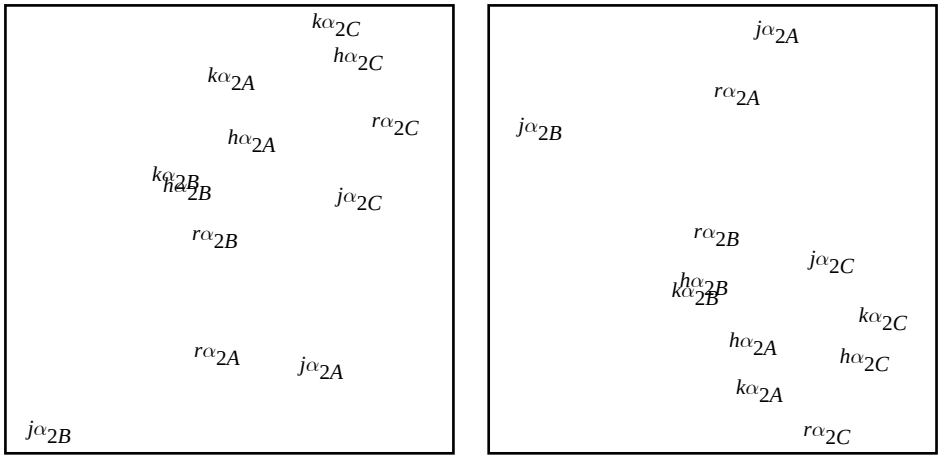


5.16 pav. Trijų žmogaus (*h*) ir penkių žuvytės (*d*) α_2-adrenoceptorinių baltymų vaizdai DS, gauti naudojantis: a) Euklido atstumais, b) miesto kvartalo atstumais

Farmakologinių duomenų [140] žmogaus, žiurkės, jūrų kiaulytės ir kiaulės α_2-adrenoceptorinių baltymų vaizdai parodyti 5.17 paveiksle. Žmogaus baltymai pažymėti pirma raide „*h*“, žiurkės – „*r*“, jūrų kiaulytės – „*j*“, o kiaulės – „*k*“. Iš paveikslo matyti glaudus $h\alpha_{2B}$, $k\alpha_{2B}$ ir $r\alpha_{2B}$ grupavimasis. Jūrų kiaulytės α_{2B}-adrenoceptorinių baltymų savybės yra gana skirtingos, palyginti su kitų rūšių. Vaizde α_{2A}-adrenoceptoriniai baltymai suformuoja du klasterius: $h\alpha_{2A}$ su $k\alpha_{2A}$ ir $r\alpha_{2A}$ su $j\alpha_{2A}$. Visų rūšių α_{2C}-adrenoceptoriniai baltymai suformuoja bendrą klasterį, bet DS su miesto kvartalo atstumais žiurkės α_{2C} vaizdas yra arčiau kiaulės α_{2A} negu kitų α_{2C} vaizdų.

DS su Euklido atstumais jūrų kiaulytės α_{2C} vaizdas yra arčiau žiurkės ir tolokai nuo kiaulės α_{2C} vaizdų, o DS su miesto kvartalo atstumais jūrų kiaulytės α_{2C} vaizdas yra arčiau kiaulės, bet tolokai nuo žiurkės α_{2C} vaizdų. Žmogaus α_2-adrenoceptorinių baltymų savybės yra panašios į kiaulės, α_{2B} ir α_{2C} – į žiurkės, bet nepanašios į jūrų kiaulytės. Taip pat matyti, kad žiurkės ir jūrų kiaulytės baltymų savybės yra gana skirtingos.

Kritinės α_1-adrenoceptorinių baltymų amino rūgštys buvo tiriamos [76], siekiant nustatyti, kurios iš jų lemia tam tikrų ligandų sąryšį su šiais baltymais. Dėl to buvo išmatuoti kelių ligandų sąryšiai su natūraliais α_{1a}- ir α_{1b}-adrenoceptoriniais baltymais ir jų atmainomis, gautomis vieną arba dvi amino rūgštis dirbtiniu mutavimu pakeitus atitinkamomis kito adrenoceptorinio baltymo rūgštimis.



a) b)

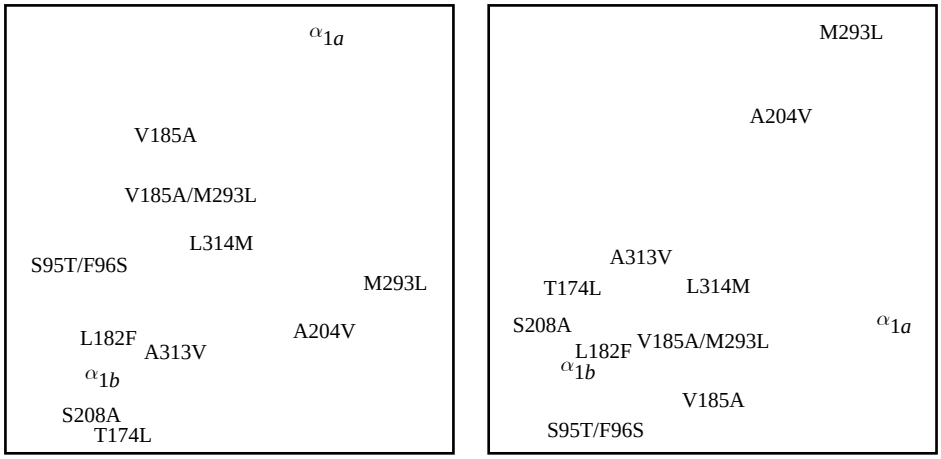
5.17 pav. Žmogaus (*h*), žiurkės (*r*), jūrų kiaulytės (*j*) ir kiaulės (*k*) α_2-adrenoceptorinių baltymų vaizdai, gauti naudojantis: a) Euklido atstumais, b) miesto kvartalo atstumais

DS su Euklido atstumais jūrų kiaulytės α_{2C} vaizdas yra arčiau žiurkės ir tolokai nuo kiaulės α_{2C} vaizdų, o DS su miesto kvartalo atstumais jūrų kiaulytės α_{2C} vaizdas yra arčiau kiaulės, bet tolokai nuo žiurkės α_{2C} vaizdų. Žmogaus α_2-adrenoceptorinių baltymų savybės yra panašios į kiaulės, α_{2B} ir α_{2C} – į žiurkės, bet nepanašios į jūrų kiaulytės. Taip pat matyti, kad žiurkės ir jūrų kiaulytės baltymų savybės yra gana skirtingos.

Kritinės α_1-adrenoceptorinių baltymų amino rūgštys buvo tiriamos [76], siekiant nustatyti, kurios iš jų lemia tam tikrų ligandų sąryšį su šiais baltymais. Dėl to buvo išmatuoti kelių ligandų sąryšiai su natūraliais α_{1a}- ir α_{1b}-adrenoceptoriniais baltymais ir jų atmainomis, gautomis vieną arba dvi amino rūgšti dirbtiniu mutavimu pakeitus atitinkamomis kito adrenoceptorinio baltymo rūgštimis.

Sąryšio duomenys buvo analizuojami ieškant, kurie pakeitimai padarė baltymo savybes panašias į kito baltymo. Analizuoti sąryšio duomenų matricą ieškant tokių pokyčių yra sudėtinga, o vizualizavimas šią paiešką gerokai palengvina.

Šių farmakologinių duomenų α_1-adrenoceptorinių baltymų ir jų mutantų vaizdai DS pateikti 5.18 paveiksle. Matyti, kad α_{1b}-adrenoceptorinio baltymo mutantų T174L, L182F ir S208A savybės yra panašios į šio baltymo. Mutacijos S95T/F96S ir A204V labai pakeičia savybes, bet nepadaro šio baltymo savybių panašių į kito baltymo.



5.18 pav. α_1-adrenoceptorinių baltymų ir jų mutantų vaizdai DS, gauti naudojantis: a) Euklido atstumais, b) miesto kvartalo atstumais

Panašiai mutacija M293L pakeičia α_{1a}-adrenoceptorinio baltymo savybes, bet nepadaro panašių į pirmojo. Mutantų L314M, V185A ir V185A/M293L savybės yra panašios, jie atvaizduoti tarp natūralių α_{1a}- ir α_{1b}-adrenoceptorinių baltymų, dėl to panašu, kad tai yra ligandų sąryšio kritinės rūgštys.

5.3. Vizualioji koreliacinių matricų analizė

Analizuokime duomenų vektorius $X_1, X_2, \ldots, X_m$, kurių kiekvienas sudarytas iš n komponenčių $x_1, x_2, \ldots, x_n$, t. y. $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $i = 1, \ldots, m$, čia m yra analizuojamų vektorių skaičius. Kiekvienas duomenų vektorius – tai tiriamos duomenų aibės objektas. Kiekviena komponentė – tai parametras ar požymis, nusakantis atskirą tiriamos duomenų aibės objektų savybę. Vieno parametro reikšmės gali priklausyti nuo kitų parametrų reikšmių, t. y. jie gali būti koreliuoti. Kyla problema, kaip nustatyti, kokie parametrai yra labiau panašūs, kurie – labiau skirtingi, kurie turi būti analizuojami kartu, o kuriuos verta atmesti, t. y. analizuojamus vektorius sudaryti tik iš dalies parametrų (komponenčių) $x_1, x_2, \ldots, x_n$. Todėl tikslinga nagrinėti vektorius sudarančių komponenčių (parametrų) rinkinį.

Vienas iš būdų – stebėti, kaip objektus nusakantys parametrai (ne patys objektai) išsidėsto plokštumoje. Galima sudaryti naują duomenų vektorių aibę,

kurios objektai yra parametrai $x_1, x_2, \ldots, x_n$, o komponentės – $X_1, X_2, \ldots, X_m$, t. y. transponuojama pradinė duomenų matrica *X*. Tačiau tokia parametrų analizė nėra efektyvi dėl kelių priežasčių. Pirma, jeigu objektų yra daug, gaunami labai didelio skaičiaus matmenų vektoriai ir juos spaudžiant į dvimatę erdvę dideli iškraipymai yra neišvengiami. Antra, $X_1, X_2, \ldots, X_m$ nėra dydžiai apibūdinantys $x_1, x_2, \ldots, x_n$.

Daug efektyvesnis būdas – nagrinėti parametrų rinkinį jų koreliacinės matricos pagrindu ir stebėti parametrų išsidėstymą plokštumoje priklausomai nuo jų koreliacijų. Tokios analizės efektyvumas eksperimentiškai pagrįstas [37], [38], [39], [45] darbuose.

5.3.1. Koreliacinių matricų vizualios analizės teorinis pagrindas

Tegul $R = \{r_{ij},\ i, j = 1, \ldots, n\}$ yra parametrų $x_1, x_2, \ldots, x_n$ koreliacinė matrica, čia r_{ij} yra parametrų x_i ir x_j koreliacijos koeficientas, skaičiuojamas pagal (2.1) formulę. Teigiama, kad parametrai x_i ir x_j yra labiau susiję (labiau koreliuoti), kai koreliacijos koeficientas absoliučiuoju didumu, t. y. $|r_{ij}|$, yra didesnis, ir mažiau susiję, kai $|r_{ij}|$ yra mažesnis. Jeigu $|r_{ij}| = 0$, sąsajos tarp parametrų nėra. Jeigu $r_{ij} = 1$ arba $r_{ij} = -1$, sąsaja tarp parametrų yra labai stipri.

Tegul S^n yra *n*-matės Euklido erdvės R^n poerdvis, sudarytas iš vienetinio ilgio vektorių, t. y. S^n yra vienetinė sfera. Jeigu $V \in S^n$, tai $\|V\| = 1$. Straipsnyje [37] pasiūlyta vektorių $V_1, V_2, \ldots, V_n \in S^n$ rinkinį, atitinkantį parametrų $x_1, x_2, \ldots, x_n$ rinkinį, apibrėžti taip, kad $\cos(V_i, V_j) = |r_{ij}|$ arba $\cos(V_i, V_j) = r_{ij}^2$.

Iš kosinusų matricos

$$\bar{K} = \{\cos(V_i, V_j),\ i, j = 1, \ldots, n\}$$

vektorius $V_s = (v_{s1}, v_{s2}, \ldots, v_{sn}) \in S^n$, $s = 1, \ldots, n$, galima atkurti naudojantis formule:

$$v_{sk} = \sqrt{\lambda_k}\, e_{ks},\ k = 1, \ldots, n,$$

čia λ_k yra matricos $\bar{K}$ *k*-oji tikrinė reikšmė, $(e_{k1}, e_{k2}, \ldots, e_{kn})$ – normalizuotas iki vienetinio ilgio (vienetinis) tikrinis vektorius, atitinkantis tikrinę reikšmę λ_k.

1 pastaba. Egzistuoja vektorių $V_1, V_2, \ldots, V_n \in S^n$ rinkinys, atitinkantis parametrų $x_1, x_2, ..., x_n$ rinkinį, jeigu tų vektorių skaliarinių sandaugų matrica yra neneigiamai apibrėžta.

Teorema. Tegul $R = \{r_{ij},\ i, j = 1, \ldots, n\}$ yra koreliacinė matrica. Matrica $R^2 = \{r_{ij}^2,\ i, j = 1, \ldots, n\}$ yra neneigiamai apibrėžta.

Įrodymas. Tegul λ_l yra matricos *R* *l*-oji tikrinė reikšmė, e_{li} – *i*-oji vienetinio matricos *R* tikrinio vektoriaus, atitinkančio tikrinę reikšmę λ_l, komponentė. Analizuokime kvadratinę formą:

$$\psi(t_1, \ldots, t_n) = \sum_{i=1}^{n} \sum_{j=1}^{n} r_{ij}^2 t_i t_j.$$

Ji gali būti pertvarkyta taip:

$$\psi(t_1, \ldots, t_n) = \sum_{i=1}^{n} \sum_{j=1}^{n} (\sum_{s=1}^{n} \lambda_s e_{si} e_{sj})^2 t_i t_j =$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} \sum_{l=1}^{n} \lambda_k \lambda_l e_{ki} e_{li} e_{kj} e_{lj} t_i t_j =$$

$$= \sum_{k=1}^{n} \sum_{l=1}^{n} \lambda_k \lambda_l (\sum_{i=1}^{n} e_{ki} e_{li} t_i)^2.$$

Koreliacinė matrica *R* yra neneigiamai apibrėžta, todėl jos tikrinės reikšmės yra neneigiamos, t. y. $\lambda_l \geq 0, l = 1, \ldots, n$ ir

$$\sum_{k=1}^{n} \sum_{l=1}^{n} \lambda_k \lambda_l (\sum_{i=1}^{n} e_{ki} e_{li} t_i)^2 \geq 0.$$

Taigi kvadratinė forma $\psi(t_1, \ldots, t_n)$ yra neneigiamai apibrėžta bet kokiam $t_1, \ldots, t_n$. Teorema įrodyta.

2 pastaba. Matricos $|R| = \{|r_{ij}|,\ i, j = 1, \ldots, n\}$ neneigiamas apibrėžtumas neišplaukia iš matricos *R* neneigiamo apibrėžtumo: matrica $|R|$ gali nebūti neneigiamai apibrėžta, jei matrica *R* turi nors vieną neigiamą elementą.

3 pastaba. Iš teoremos ir 2 pastabos gauname, kad egzistuoja vektorių $V_1, V_2, \ldots, V_n \in S^n$ rinkinys, atitinkantis parametrų $x_1, x_2, \ldots, x_n$ rinkinį, jeigu:

a) $\cos(V_i, V_j) = r_{ij},\ i, j = 1, \ldots, n,$ kai visi r_{ij} nėra neigiami, t. y. $r_{ij} \geq 0$;

b) $\cos(V_i,V_j)=r_{ij}^2$, $i,j=1,\ldots,n$, kai yra nors vienas neigiamas r_{ij}, t. y. $r_{ij}<0$.

4 pastaba. Vektorių $V_1,V_2,\ldots,V_n\in S^n$ rinkinį, atitinkantį $x_1,x_2,\ldots,x_n$ rinkinį, atvaizdavus plokštumoje taip, kad būtų išlaikyti santykiniai atstumai tarp $V_1,V_2,\ldots,V_n\in S^n$, galima vizualiai stebėti parametrų $x_1,x_2,\ldots,x_n$ išsidėstymą plokštumoje.

Tolesniuose skyreliuose supažindinama su koreliacinių matricų vizualiosios analizės taikymais sprendžiant realius uždavinius. Oftalmologinių parametrų sistemos tyrimas koreliacinės matricos pagrindu jau buvo pateiktas 5.2.1 skyrelyje.

5.3.2. Psichologiniai testai

Yra testuoti 145 VII ir VIII klasių Čikagos mokiniai. Viso – 24 testai. Pažymėkime juos $x_1,x_2,\ldots,x_{24}$. Kiekvieną mokinį apibūdina 24 skaičių rinkinys (atsakymų į kiekvieną testą įverčiai). Faktiškai gauta lentelė, kurioje yra 145 eilutės ir 24 stulpeliai, t. y. tiriamą duomenų aibę sudaro 145 objektai (mokiniai) ir 24 juos charakterizuojantys parametrai $x_1,x_2,\ldots,x_{24}$ (atitinkamų testų rezultatai). Pagal mokinių atsakymus į testus buvo apskaičiuota testų koreliacinė matrica. Uždavinys yra įvertinti testų artimumus jų koreliacinės matricos pagrindu.

Penkios testų grupės įvardintos [67] darbe:

1) erdvės suvokimo (*spatial perception*) – $\{x_1,\ldots,x_4\}$;

2) žodiniai (*verbal test*) – $\{x_5,\ldots,x_9\}$;

3) mąstymo spartos (*rapidity of thinking*) – $\{x_{10},\ldots,x_{13}\}$;

4) atminties (*memory*) – $\{x_{14},\ldots,x_{19}\}$;

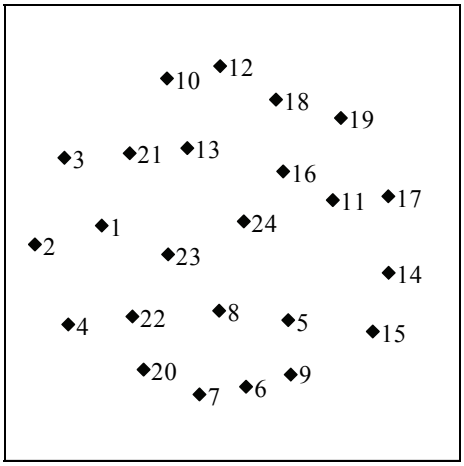
5) matematinių gebėjimų (*mathematical capabilities*) – $\{x_{20},\ldots,x_{24}\}$.

Šių penkių grupių testai apibūdina bendrą testuojamo asmens išsivystymą. Tyrimai [36] parodė, kad faktiškai yra keturios testų grupės, o penktosios grupės testai pasiskirsto tarp kitų keturių grupių:

$\{x_1,\ldots,x_4,x_{20},x_{22},x_{23}\}$, $\{x_5,\ldots,x_9\}$, $\{x_{10},\ldots,x_{13},x_{21},x_{24}\}$, $\{x_{14},\ldots,x_{19}\}$.

Pasitelksime testų $x_1,x_2,\ldots,x_{24}$ tyrimui vizualiąją analizę. Iš jų koreliacinės matricos, remiantis 5.3.1 skyrelyje aprašytu metodu, gauti 24-mačiai vektoriai. Vienas analizuojamas vektorius atitinka vieną iš 24 testų $\{x_1,x_2,\ldots,x_{24}\}$. Iš pradžių šiuos vektorius atvaizduokime plokštumoje pagal

Sammono algoritmą (žr. 5.19 pav.). Paveiksle pažymėti skaičiai atitinka testų $\{x_1, x_2, \ldots, x_{24}\}$ numerius. Čia jokių susidariusių grupių (klasterių) įžvelgti neįmanoma, visi taškai išsidėstę gana tolygiai, daryti išvadas apie duomenų klasterius yra sunku. Šis eksperimentas parodo, kad kartais klasteriams duomenyse nustatyti Sammono projekcijos (kaip ir kitų DS grupės algoritmų) nepakanka. Žinant, kad nagrinėjami psichologiniai testai turėtų sudaryti tam tikras grupes, reikia ieškoti alternatyvių vizualizavimo būdų, kurie parodytų šias grupes. Išmokykime SOM tinklą vektoriais, apskaičiuotais iš psichologinių testų $\{x_1, x_2, \ldots, x_{24}\}$ koreliacinės matricos. Analizuojami vektoriai, o tuo pačiu ir patys testai, pasiskirsto tarp tinklo langelių. Eksperimentuose naudoti [3x3] ir [4x4] dydžio SOM tinklai (žr. 5.7 ir 5.8 lenteles). Lentelėse pažymėti analizuojamų testų $\{x_1, x_2, \ldots, x_{24}\}$, pakliuvusių į tam tikrus langelius, numeriai. Matyti, kaip SOM tinklas klasterizuoja duomenis. Iš 5.7 lentelės jau galima matyti susidariusius keturis klasterius, tačiau 5.8 lentelei reikia papildomos analizės, kadangi sunku spręsti, ar į gretimus langelius pakliuvę testai sudaro atskirus klasterius ir kaip arti jie yra vienas nuo kito.



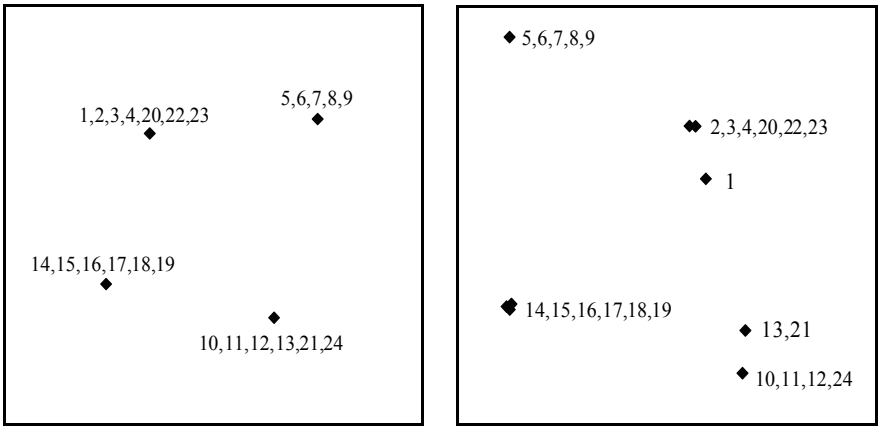
5.19 pav. Sammono metodu vizualizuoti psichologiniai testai

5.7 lentelė. Psichologinių testų išsidėstymas SOM [3x3] tinkle

14-19		10-13, 21, 24
1-4, 20, 22, 23		5-9

5.8 lentelė. Psichologinių testų išsidėstymas SOM [4x4] tinkle

10, 11, 12, 24		17, 18	14, 15, 16
13, 21		19	
1			
2, 3, 4	20, 22, 23		5, 6, 7, 8, 9



a) b)

5.20 pav. Psichologinių testų išdėstymas plokštumoje, gautas naudojantis SOM tinklo ir Sammono projekcijos junginiu: a) SOM [3x3], b) SOM [4x4]

Taškų išsidėstymas plokštumoje, kai po SOM tinklo mokymo gauti vektoriai nugalėtojai vizualizuojami Sammono algoritmu, parodytas 5.20 paveiksle. Čia jau pastebimi susidarę ryškūs klasteriai. Galima vizualiai nustatyti, kurios testų grupės yra labai artimos, kurios – mažiau. Iš 5.8 lentelės buvo galima manyti, kad testai x_1, x_{13}, x_{21} yra panašūs. Tačiau 5.20b paveikslas paneigia tokį tvirtinimą: testai x_{13}, x_{21} yra iš tikrųjų panašūs (artimi), tačiau testas x_1 yra toli ir nuo x_{13} ir nuo x_{21}, jis yra arčiau kitos grupės, t. y. testų $x_2 - x_4$, x_{20}, x_{22}, x_{23}.

Taigi galima padaryti bendras išvadas apie psichologinių testų grupes. Aiškiai matomi keturi klasteriai pagal atskiras testų grupes: erdvės suvokimo $\{x_1,\ldots,x_4\}$, žodinių $\{x_5,\ldots,x_9\}$, mąstymo spartos $\{x_{10},\ldots,x_{13}\}$, atminties $\{x_{14},\ldots,x_{19}\}$. Kai kurie matematinių gebėjimų grupės testai $\{x_{20},x_{22},x_{23}\}$ „prilimpa“ prie erdvės suvokimo testų, o testai $\{x_{21},x_{24}\}$ – prie mąstymo

spartos testų. Tačiau tai pamatyti vien iš Sammono algoritmo rezultatų (žr. 5.19 pav.) ar vien tik iš SOM tinklo (5.7 lentelių) būtų neįmanoma. Tai svarus argumentas, kodėl tikslinga naudoti SOM tinklo ir Sammono algoritmo junginį, o ne tik atskirai paimtus algoritmus. Junginyje jie kompensuoja vienas kito trūkumus.

5.3.3. Meteorologiniai parametrai

Buvo išmatuoti šie meteorologiniai parametrai, aprašantys oro užterštumą Vilniuje [153]:

x_1, x_2, x_3 – anglies monoksido, azoto oksido ir ozono koncentracijos,

x_4 – vertikalus temperatūros gradientas, išmatuotas 2–8 metrų aukštyje,

x_5 – saulės radiacijos intensyvumas,

x_6 – atmosferos paribio storis,

x_7 – kritulių kiekis,

x_8 – temperatūra,

x_9 – vėjo greitis,

x_{10} – atmosferos stabilumo klasė.

Iš parametrų $x_1, x_2, \ldots, x_{10}$ koreliacinės matricos, remiantis 5.3.1 skyrelyje aprašytu metodu, gauti 10-mačiai vektoriai. Vienas analizuojamas vektorius atitinka vieną iš 10 parametrų $x_1, x_2, \ldots, x_{10}$.

Meteorologinių parametrų išsidėstymas [4x4] dydžio SOM tinkle parodytas 5.9 lentelėje. Langeliai, atitinkantys neuronus nugalėtojus, užpildomi parametrų $x_1, x_2, ..., x_{10}$ numeriais, keli langeliai lieka tušti. Galime teigti, kad susidaro šios parametrų grupės: $\{x_5, x_{10}\}$, $\{x_3, x_6, x_9\}$ ir $\{x_1, x_2, x_4, x_8\}$, parametras x_7 šiek tiek nutolęs nuo paskutinės grupės. Norėdami pagrįsti ar paneigti šį teiginį, po SOM tinklo mokymo gautus neuronus nugalėtojus vizualizuokime Sammono algoritmu. Gausime parametrų išsidėstymą plokštumoje, kuris yra informatyvesnis nei jų išsidėstymas lentelėje.

Rezultatai, gauti SOM [4x4] tinklo ir Sammono projekcijos junginiu, pavaizduoti 5.21 paveiksle. Čia matome, kad parametras x_8 yra gana toli nuo parametrų grupės $\{x_1, x_2, x_4\}$ nors SOM tinkle šis parametras ir minėta grupė buvo gretimuose lentelės langeliuose. Paveiksle parametras x_8 sudaro atskirą grupę su parametru x_7.

Detalūs tyrimų rezultatai pateikti [38] straipsnyje.

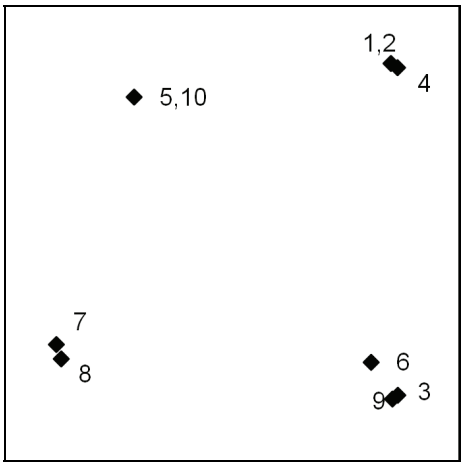
Galime daryti išvadas, kad analizuojami meteorologiniai parametrai sudaro šias grupes:

- $\{x_1, x_2, x_4\}$ – anglies monoksido, azoto oksido koncentracijos, vertikalus temperatūros gradientas;
- $\{x_3, x_6, x_9\}$ – ozono koncentracija, atmosferos paribio storis, vėjo greitis;
- $\{x_7, x_8\}$ – kritulių kiekis, temperatūra;
- $\{x_5, x_{10}\}$ – saulės radiacijos intensyvumas, atmosferos stabilumo klasė.

Kaip matome, dažniausiai grupes sudaro pagal prasmę artimi parametrai. Pavyzdžiui, ozono koncentracija ir atmosferos paribio storis, saulės radiacijos intensyvumas ir atmosferos stabilumo klasė ar anglies monoksido ir azoto oksido koncentracijos. Reikia atkreipti dėmesį į parametrus, kurie grupuojasi su prasmingai artimais parametrais ir ieškoti to grupavimosi teorinio pagrindimo.

5.9 lentelė. Meteorologinių parametrų išsidėstymas SOM [4x4] tinkle

5, 10		7	
			8
6			4
3	9		1, 2



5.21 pav. SOM tinklo ir Sammono projekcijos junginiu vizualizuoti meteorologiniai parametrai

5.3.4. Kopas apibūdinantys parametrai

Suomijos pajūrio kopas ir jų vegetaciją apibūdina šie parametrai [71]:

x_1 – atstumas nuo kranto,

x_2 – aukštis virš jūros lygio,

x_3 – dirvožemio PH,

x_4, x_5, x_6, x_7 – kalcio, fosforo, kalio, mangano kiekis,

x_8, x_9 – vidutinis smėlio skersmuo ir jo rūšis,

x_{10} – šiaurumas pagal suomišką koordinačių sistemą,

x_{11} – žemės antsprūdžio laipsnis,

x_{12} – jūros lygio svyravimas,

x_{13} – dirvožemio drėgnumas,

x_{14} – šlaito tangentas,

x_{15} – smėlio paviršiaus dalis,

x_{16} – medžiais apaugusi dalis.

Iš parametrų $x_1, x_2, \ldots, x_{16}$ koreliacinės matricos, remiantis 5.3.1 skyrelyje aprašytu metodu, gauti 16-mačiai vektoriai. Vienas analizuojamas vektorius atitinka vieną iš 16 parametrų $x_1, x_2, \ldots, x_{16}$. Įdomu būtų pažiūrėti, ar šie parametrai sudaro grupes, o jei sudaro, tai kokias. Tam tikslui naudotasi SOM tinklu ir jo junginiu su Sammono algoritmu.

Suomijos pajūrio kopas ir jų vegetaciją apibūdinančių parametrų išsidėstymas [4x4] dydžio SOM tinkle parodytas 5.10 lentelėje. Langeliai, atitinkantys neuronus nugalėtojus, užpildomi parametrų $x_1, x_2, \ldots, x_{16}$ numeriais, keli langeliai lieka tušti. Čia tam tikrą parametrų grupavimąsi galima įžvelgti. Yra bent trys grupės, kurias skiria tušti lentelės langeliai. Tačiau lieka neaišku, ar parametrai $x_4, x_6, x_7, x_3, x_1, x_2, x_{16}$ sudaro vieną grupę ar kelias, kaip toli yra parametras x_{13} nuo parametrų x_{10}, x_{11}, x_{12} grupės, ar parametrai $x_5, x_8, x_9, x_{14}, x_{15}$ sudaro vieną grupę.

5.10 lentelė. Kopas apibūdinančių parametrų išsidėstymas SOM [4x4] tinkle

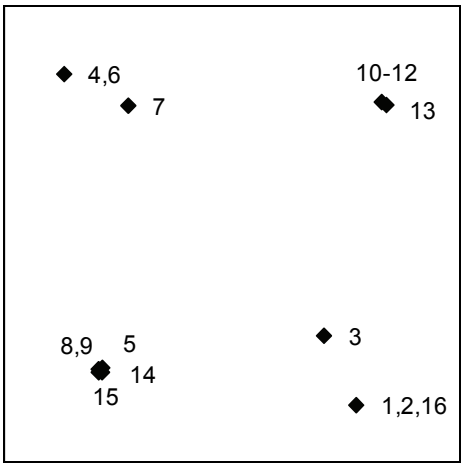
4, 6		5	8, 9
7		15	14
3			
1, 2, 16		13	10, 11, 12

Atsakymus į šiuos klausimus galima rasti, parametrus $x_1, x_2, \ldots, x_{16}$ atvaizdavus SOM tinklo ir Sammono algoritmo junginiu (žr. 5.22 pav.). Čia matome, kad parametrai $x_4, x_6, x_7, x_3, x_1, x_2, x_{16}$ sudaro du klasterius, o ne vieną, parametras x_{13} yra visai netoli nuo parametrų x_{10}, x_{11}, x_{12} grupės, parametrai $x_5, x_8, x_9, x_{14}, x_{15}$ sudaro tik vieną klasterį. Detalūs tyrimų rezultatai pateikti [38] straipsnyje.

Taigi analizuojami kopas apibūdinantys parametrai sudaro šiuos klasterius (grupes):

- $\{x_1, x_2, x_3, x_{16}\}$ – atstumas nuo kranto, aukštis virš jūros lygio, dirvožemio PH, medžiais apaugusi dalis;
- $\{x_4, x_6, x_7\}$ – kalcio, kalio, mangano kiekis;
- $\{x_5, x_8, x_9, x_{14}, x_{15}\}$ – fosforo kiekis, vidutinis smėlio skersmuo ir jo rūšis, šlaito tangentas, smėlio paviršiaus dalis;
- $\{x_{10}, x_{11}, x_{12}, x_{13}\}$ – šiaurumas pagal suomišką koordinačių sistemą, žemės kilimo greitis, jūros lygio svyravimas, dirvožemio drėgnumas.

Kaip ir analizuojant meteorologinius parametrus, grupes sudaro pagal prasmę artimi parametrai. Pavyzdžiui, vidutinis smėlio skersmuo, jo rūšis ir smėlio paviršiaus dalis, atstumas nuo kranto ir medžiais apaugusi dalis. Į kitų parametrų priklausomybę grupėms turėtų atkreipti dėmesį ekologai bei aiškintis to priežastis.



5.22 pav. SOM tinklo ir Sammono projekcijos junginiu vizualizuoti kopas apibūdinantys parametrai

5.3.5. Studijų programos

Dažnai studentai iš susijusių studijuojamų dalykų gauna panašius pažymius. Jei studentas yra gabus humanitariniams mokslams, jam humanitarinių mokslų dalykai sekasi. Gabūs matematikai gauna gerus matematinių dalykų įvertinimus. Dažniausiai iš anksto žinome, kurie dalykai yra matematiniai, kurie – humanitariniai. Tačiau yra tokių, kurie negali būti vienareikšmiškai priskiriami prie vienos iš gerai žinomų dalykų klasės, pavyzdžiui, informatikos dalykai.

Atlikta analizė padeda įvertinti skirtingų informatikos dalykų matematizavimo lygį ar jų artimumą humanitariniams dalykams. Sunku įvertinti kiekybiškai nagrinėjant atskirus studijų dalykus, tačiau tai galima padaryti atsižvelgiant į visų studijų dalykų, sudarančių studijų programą, visumą [39].

Analizuojamus duomenis sudaro Vilniaus pedagoginio universiteto matematikos ir informatikos fakulteto 41 studento, sėkmingai baigusio bakalauro studijas, studijų metu išlaikytų 25 egzaminų rezultatai. Analizuojami šie dalykai:

x_1 – tikimybių teorija,
x_2 – matematika ir jos dėstymo metodika,
x_3 – geometrija 1,
x_4 – pedagogika ir psichologija,
x_5 – geometrija 2,
x_6 – matematinė analizė 1,
x_7 – pedagogika,
x_8 – geometrija 3,
x_9 – psichologija 1,
x_{10} – algebra,
x_{11} – matematinė analizė 2,
x_{12} – užsienio kalba,
x_{13} – geometrija 4,
x_{14} – psichologija 2,
x_{15} – algebra ir skaičių teorija 1,
x_{16} – matematinė analizė 3,
x_{17} – informatika 1,

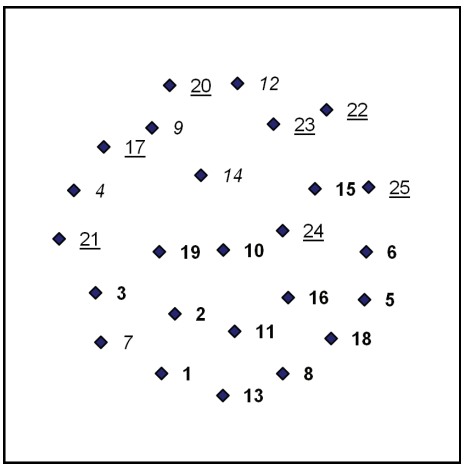
x_{18} – algebra ir skaičių teorija 2,
x_{19} – matematinė analizė 4,
x_{20} – informatika 2,
x_{21} – mokomųjų programų kūrimas,
x_{22} – informatikos dėstymo teorija,
x_{23} – matematinių uždavinių sprendimo paketai,
x_{24} – algoritmų teorija,
x_{25} – programavimo metodai.

Kai kurių studijų dalykų egzaminai buvo laikomi keliuose semestruose. Skaičius po tokio dalyko pavadinimo rodo egzamino eiliškumą.

Studijų dalykus galima suskirstyti į tris grupes:

- matematiniai: $x_1 - x_3, x_5, x_6, x_{10}, x_{11}, x_{13}, x_{15}, x_{16}, x_{18}, x_{19}$;
- humanitariniai: $x_4, x_7, x_9, x_{12}, x_{14}$;
- informatikos: $x_{17}, x_{20} - x_{25}$.

Kiekvieną studentą apibūdina 25 skaičių rinkinys (egzaminų pažymiai). Faktiškai gauta lentelė, kurioje yra 41 eilutės ir 25 stulpeliai, t. y. tiriamą duomenų aibę sudaro 41 objektai (studentai) ir 25 juos charakterizuojantys parametrai $x_1, x_2, \ldots, x_{25}$ (atitinkamų egzaminų pažymiai). Pagal egzaminų rezultatus buvo apskaičiuota studijų dalykų koreliacinė matrica. Uždavinys yra įvertinti studijų dalykų artimumus jų koreliacinės matricos pagrindu.



5.23 pav. Sammono algoritmu vizualizuoti studijų dalykai

Iš dalykų $x_1, x_2, \ldots, x_{25}$, atitinkančių egzaminų įvertinimus, koreliacinės matricos, naudojantis 5.3.1 skyrelyje aprašytu metodu, gauti 25-mačiai vektoriai. Vienas analizuojamas vektorius atitinka vieną iš 25 dalykų $x_1, x_2, \ldots, x_{25}$.

Sammono algoritmu vizualizuoti studijų dalykus atitinkantys 25-mačiai vektoriai pavaizduoti 5.23 paveiksle. Faktiškai čia matome studijų dalykų išsidėstymą plokštumoje. Skaičiai nurodo dalykų $x_1, x_2, \ldots, x_{25}$ numerius. Matematinių dalykų numeriai yra paryškinti, informatikos – pabraukti, o humanitarinių dalykų – pasvirę. Nors dalykai yra išsidėstę gana tolygiai, tačiau tam tikrus jų panašumus galima įžvelgti.

Duomenis vizualizuojant SOM tinklu, jo mokymui buvo naudotas skirtingas epochų skaičius *e*. Epocha – tai mokymo proceso dalis, kurios metu visi mokymo aibės vektoriai po vieną kartą pateikiami į tinklą. Vizualizavimo rezultatai pateikti 5.11 ($e = 200$) ir 5.12 ($e = 5000$) lentelėse. Čia SOM tinklo dydis yra [4x4].

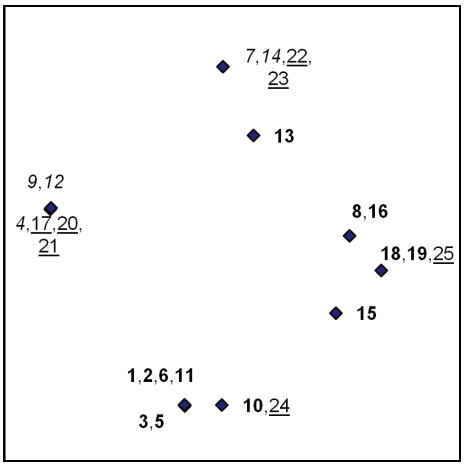
Palyginus su Sammono algoritmu, SOM tinkle duomenys yra susigrupavę (žr. 5.11 ir 5.12 lenteles), tačiau čia dar lieka nemažai neaiškumų, pavyzdžiui, ar dalykas x_{13} yra artimesnis dalykams x_8 ir x_{16}, ar dalykams x_7, x_{14}, x_{22} ir x_{23}. Antra vertus, didėjant tinklo mokymo epochų skaičiui kito dalykų x_9 ir x_{12} vieta – pradžioje jie buvo skirtinguose, bet gretimuose lentelės langeliuose, vėliau pateko į vieną. Tai reiškia, kad nėra stipraus ryšio tarp šių dviejų parametrų, nes tinklo mokymo metu vėlesnėse epochose tik jų vieta kito.

5.11 lentelė. Studijų dalykų išsidėstymas SOM tinkle ($e = 200$)

1, **2**, **6**, **11**	**3**, **5**		*4*, 17, 20, 21
10, 24			*9*
15			*12*
18, **19**, 25	**8**, **16**	**13**	*7*, *14*, 22, 23

5.12 lentelė. Studijų dalykų išsidėstymas SOM tinkle ($e = 5000$)

4, 17, 20, 21		**3**, **5**	**1**, **2**, **6**, **11**
9, *12*			**10**, 24
			15
7, *14*, 22, 23	**13**	**8**, **16**	**18**, **19**, 25



5.24 pav. SOM tinklo ir Sammono algoritmo junginiu vizualizuoti studijų dalykai

Aiškesni rezultatai gaunami vizualizavus duomenis SOM tinklo ir Sammono algoritmo junginiu (žr. 5.24 pav., $e = 5000$). Čia matome ne tik studijų dalykų grupes, bet ir jų tarpusavio panašumus. Pavyzdžiui, dalykas x_{13} yra artimesnis dalykams x_7, x_{14}, x_{22} ir x_{23} nei x_8 ir x_{16}, o tai buvo neįmanoma sužinoti iš 5.11 ar 5.12 lentelių.

Iš gautų rezultatų galime daryti tokias išvadas:

- matematiniai ir humanitariniai dalykai turi tendenciją pasidalyti į skirtingas grupes ir tik vienintelis matematinis dalykas (geometrija 4 (x_{13})) yra arčiau humanitarinių dalykų;
- visi informatikos dalykai nesuformuoja atskiros grupės; dalis informatikos dalykų patenka į matematinių dalykų grupę, kita dalis – į humanitarinių;
- du informatikos dalykai (algoritmų teorija (x_{24}) ir programavimo metodai (x_{25})) yra gana matematizuoti; kiti informatikos dalykai (informatika 1 (x_{17}), informatika 2 (x_{20}), mokomųjų programų kūrimas (x_{21}), informatikos dėstymo teorija (x_{22}) ir matematinių uždavinių sprendimo paketai (x_{23})) turi panašumų su humanitariniais dalykais.

Studijų dalykų tyrimas, kaip ir anksčiau aptarti kiti taikymai naudojantis vizualiąja analize, parodė žmogaus galimybes nugalėti duomenų daugiamatiškumą ir priimti teisingus sprendimus.

Baigiamasis žodis

Daugiamačių duomenų vizualizavimas yra svarbus duomenų analizės įrankis, padedantis geriau suvokti daugiamačių duomenų struktūrą – susidariusias grupes (klasterius), labai išsiskiriančius objektus (taškus atsiskyrėlius), objektų tarpusavio artimumą jų visumos kontekste. Ir čia esminis vaidmuo tenka žmogui, kuris daro išvadas ir priima galutinį sprendimą.

Vadovėlį sudaro penki pagrindiniai skyriai: *Įvadas*, *Daugiamačių duomenų vizualizavimo strategijos*, *Daugiamatės skalės*, *Dirbtiniai neuroniniai tinklai duomenims vizualizuoti*, *Vizualiosios analizės taikymai*. Vadovėlio medžiaga išties plataus spektro: tiesioginio vizualizavimo metodai (geometriniai, simboliniai, hierarchinio vizualizavimo), tiesinės ir netiesinės projekcijos metodai; daugiamačių skalių metodas ir minimizavimo algoritmai daugiamatėms skalėms; dirbtinių neuroninių tinklų pagrindai, saviorganizuojantys neuroniniai tinklai, SOM tinklo ir Sammono algoritmo jungimo būdai, kreivinių komponenčių analizė, daugiamatės skalės taikant dirbtinius neuroninius tinklus, autoasociatyvieji neuroniniai tinklai, neuroninių skalių metodas; socialinių duomenų analizė, taikymai medicinoje ir farmakologijoje, vizualioji koreliacinių matricų analizė.

Vadovėlis vertingas tuo, kad atskleidžiama daugiamačių duomenų vizualizavimo galimybių visuma – metodai grindžiami skirtingomis idėjomis, yra universalūs ir orientuoti specialioms duomenų struktūroms. Ne mažiau svarbu yra tai, kad čia parodyta praktinė vizualizavimo reikšmė įvairiose srityse. Socialiniuose moksluose – bendrojo lavinimo mokyklų, Centrinės Europos valstybių ekonominės ir socialinės būklės ir Seimo narių balsavimų duomenų vizualioji analizė. Biomedicinoje – oftalmologinių, fiziologinių, širdies ritmo ir farmakologinių duomenų vizualioji analizė. Atskira taikymų grupė – koreliacinės matricos. Čia vizualizuoti akies dugno nuotraukas apibūdinančių parametrų, psichologinių testų duomenys, meteorologiniai duomenys, duomenys apie kopų būklę, studijų programas.

Vadovėlio tikslai – pateikti daugiamačių duomenų vizualizavimo teoriją, išdėstyti duomenų vizualizavimo strategijas, nurodyti praktinių taikymų pavyzdžių. Jis padės populiarinti daugiamačių duomenų vizualiąją analizę tarp įvairių sričių specialistų, nes tai leidžia savo akimis pamatyti, kas yra bendru atveju sunkiai žmogui suvokiama, suteikia taip trokštamą pasižvalgymo po daugiamatę erdvę galimybę.

Literatūra

1. D. K. Agrafiotis, V. S. Lobanov (2000). Nonlinear mapping networks. *Journal of Chemical Information and Computer Science*, 40, 1356–1362.
2. A. Ahalt, A. K. Krishnamurthy, P. Chen, D. E. Melton (1990). Competitive learning algorithm for vector quantization. *Neural Networks*, 3, 277–290.
3. D. F. Andrews (1972). Plots of high dimensional data. *Biometrics*, 28, 125–136.
4. P. Arabie (1991). Was Euclid an unnecessarily sophisticated psychologist? *Psychometrika*, 56(4), 567–587.
5. P. Baldi, H. Hornik (1989). Neural networks and principal component analysis: learning from examples without local minima. *IEEE Trans. Neural Networks*, 2, 53–58.
6. R. A. Becker, W. S. Cleveland (1996). The design and control of Trellis display. *Journal of Computational and Statistical Graphics*, 5, 123–155.
7. M. Belkin, P. Niyogi (2001). Laplacian eigenmaps and spectral techniques for embedding and clustering. In: T. G. Dietterich, S. Becker, Z. Ghahramani (Eds.), *Advances in Neural Information Processing Systems*, Vol. 14. MIT Press, Cambridge, MA, pp. 585–591.
8. M. Belkin, P. Niyogi (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computing*, 15(6), 1373–1396.
9. J. Bernatavičienė, G. Dzemyda, O. Kurasova, V. Marcinkevičius (2006). Decision support for preliminary medical diagnosis integrating the data mining methods. In: H. Pranevičius, O. Vaarmann, E. Zavadskas (Eds.), *Proceedings of 5th International Conference on Simulation and Optimisation in Business and Industry*. Technologija, Kaunas, pp. 155–160.
10. J. Bernatavičienė, G. Dzemyda, O. Kurasova, V. Marcinkevičius (2006). Optimal decisions in combining the SOM with nonlinear projection methods. *European Journal of Operational Research*, 173, 729–745.

11. J. Bernatavičienė, G. Dzemyda, O. Kurasova, V. Marcinkevičius (2007). The problem of visual analysis of multidimensional medical data. In: A. Törn, J. Žilinskas (Eds.), *Models and Algorithms for Global Optimization*. *Springer Optimization and its Applications*, Vol. 4. Springer, pp. 277–298.

12. J. Bertin (1983). *Semiology of Graphics*. W. J. Berg (translator). Semiologie Graphique (Editions Gauthier-Villars). The University of Wisconsin Press, Madison, WI.

13. G. Biswas, A. A. Jain, R. C. Dubes (1981). Evaluation of projection algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3(6), 701–708.

14. I. Borg, P. Groenen (2005). *Modern Multidimensional Scaling*, 2nd ed. Springer, New York.

15. C. Brunsdon, A.S. Fotheringham, M.E. Charlton (1998). An investigation of methods for visualising highly multivariate datasets in case studies of visualization in the social sciences. In: D. Unwin, P. Fisher (Eds.), *Joint Information Systems Committee, ESRC, Technical Report Series*, Vol. 43, pp. 55–80.

16. M. J. Brusco (2001). A simulated annealing heuristics for unidimensional and multidimensional (city block) scaling of symmetric proximity matrices. *Journal of Classification*, 18(1), 3–33.

17. M. J. Brusco, S. Stahl (2005). *Branch-and-Bound Applications in Combinatorial Data Analysis*. Springer.

18. E. Cantú-Paz (2000). *Efficient and Accurate Parallel Genetic Algorithms*. Kluwer Academic Publishers.

19. Z. Chen, P. C. Ivanov, K. Hu, H. E. Stanley (2002). Effect of nonstationarities on detrended fluctuation analyzis. *Physical Review E*, 65(4), 041107-(1–15).

20. H. Chernoff (1973). The use of faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association*, 68, 361–368.

21. T. Cox, M. Cox (2001). *Multidimensional Scaling*. Chapman and Hall/CRC, Boca Raton.

22. J. De Leeuw (1977). Applications of convex analysis to multidimensional scaling. In: J. R. Barra, F. Brodeau, G. Romier, B. Van Cutsem (Eds.), *Recent Developments in Statistics*. North-Holland, pp. 133–145.

23. J. De Leeuw (1984). Differentiability of Kruskal's stress at a local minimum. *Psychometrika*, 49(1), 111–113.

24. J. De Leeuw (1988). Convergence of the majorization method for multidimensional scaling. *Journal of Classification*, 5, 163–180.

25. J. De Leeuw, W. Heiser (1982). Theory of multidimensional scaling. In: P. R. Krishnaiah, L. N. Kanal (Eds.), *Handbook of Statistics 2: Classification, Pattern Recognition and Reduction of Dimensionality*. North-Holland, pp. 285–316.

26. P. Delicado (2001). Another look at principal curves and surfaces. *Journal of Multivariate Analysis*, 77, 84–116.

27. P. Demartines, J. Hérault (1997). Curvilinear component analysis: a self-organizing neural network for nonlinear mapping of data sets. *IEEE Transaction on Neural Networks*, 8(1), 148–154.

28. D. DeMers, G. Cottrell (1993). Non-linear dimensionality reduction. In: C. G. Hanson (Eds.), *Advances in Neural Information Processing Systems*, Vol. 5. Morgan Kaufmann, San Mateo, pp. 580–587.

29. D. De Ridder, R.P.W. Duin (1997). Sammon's mapping using neural networks: a comparison. *Pattern Recognition Letters*, 18, 1307–1316.

30. W. S. De Sarbo, B. Kim, M. Wedel, D. K. H. Fong (1998). A Bayesian approach to the spatial representation of market structure from consumer choice data. *European Journal of Operational Research*, 111, 285–305.

31. V. De Silva, J. B. Tenenbaum (2003). Unsupervised learning of curved manifolds. In: D. D. Denison, M. H. Hansen, C. C. Holmes, B. Mallick, B. Yu (Eds.), *Nonlinear Estimation and Classification, Lecture Notes in Statistics*, Vol. 171. Springer-Verlag, New York, pp. 453–466.

32. E. W. Dijkstra (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1, 269–271.

33. R. O. Duda, P. E. Hart (1973). *Pattern Recognition and Scene Analysis*. John Wiley.

34. R. O. Duda, P. E. Hart, D. G. Stork (2000). *Pattern Classification*. 2nd ed. John Wiley.

35. M. H. Dunham (2003). *Data Mining Introductory and Advanced Topics*. Pearson Education, Inc. Prentice Hall.

36. G. Dzemyda (1996). Clustering of parameters on the basis of correlations via simulated annealing. *Control and Cybernetics*, 25(1), 55–74.

37. G. Dzemyda (2001). Visualization of a set of parameters characterized by their correlation matrix. *Computational Statistics and Data Analysis*, 36(10), 15–30.

38. G. Dzemyda (2004). Visualization of correlation-based environmental data. *Environmetrics*, 15, 827–836.

39. G. Dzemyda (2005). Multidimensional data visualization in the statistical analysis of curricula. *Computational Statistics and Data Analysis*, 49, 265–281.

40. G. Dzemyda, P. Gudynas, V. Šaltenis, V. Tiešis (2001). *Lietuvos pedagogai ir moksleiviai: analizė ir prognozė*. Mokslo Aidai, Vilnius.

41. G. Dzemyda, O. Kurasova (2002). Comparative analysis of the graphical result presentation in the SOM software. *Informatica*, 13(3), 275–286.

42. G. Dzemyda, O. Kurasova (2002). Dirbtinių neuroninių tinklų taikymas bendrojo lavinimo mokykloms palyginti. *Informacijos mokslai*, 22, 42–52.

43. G. Dzemyda, O. Kurasova (2005). Oftalmologinių duomenų vizuali analizė. *Informacijos mokslai*, 34, 237–242.

44. G. Dzemyda, O. Kurasova (2006). Heuristic approach for minimizing the projection error in the integrated mapping. *European Journal of Operational Research*, 171(3), 859–878.

45. G. Dzemyda, O. Kurasova (2007). Dimensionality problem in the visualization of correlation-based data. In: B. Beliczynski, A. Dzielinski, M. Iwanowski, B. Ribeiro (Eds.), *Adaptive and Natural Computing Algorithms – ICANNGA 2007*. *Lecture Notes in Computer Science*, Vol. 4432. Springer, pp. 544–553.

46. G. Dzemyda, O. Kurasova, V. Medvedev (2007). Dimension reduction and data visualization using neural networks. In: I. Maglogiannis, K. Karpouzis, M. Wallace and J. Soldatos (Eds.), *Emerging Artificial Intelligence Applications in Computer Engineering. Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies*. *Frontiers in Artificial Intelligence and Applications*, Vol. 160. IOS Press, pp. 25–49.

47. G. Dzemyda, O. Kurasova, A. Vainoras (2007). Parameter system for human physiological data representation and analysis. In: J. Marti, J. M. Benedi, A. M. Mendonca, J. Serrat (Eds.). *Pattern Recognition and Image Analysis – IbPRIA 2007*. *Lecture Notes in Computer Science*, Vol. 4477. Springer, pp. 209–216.

48. G. Dzemyda, V. Tiešis (2001). Visualization of multidimensional objects and the socio-economical impact to activity in EC RTD databases. *Informatica*, 12(2), 239–262.

49. D. S. Ebert, R. M. Rohrer, Ch. D. Shaw, P. Panda, J. M. Kukla, D. A. Roberts (2000). Procedural shape generation for multi-dimensional data visualization. *Computers & Graphics*, 24(3), 375–384.

50. J. E. Everett (2001). Algorithms for multidimensional scaling. In: L. D. Chambers (Ed.), *The Practical Handbook of Genetic Algorithms*, 2nd ed. Chapman & Hall/CRC, pp. 203–233.

51. R. A. Fisher (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7, 179–188.

52. A. Flexer (2001). On the use of self-organizing maps for clustering and visualization. *Intelligent Data Analysis*, 5, 373–84.

53. J. H. Friedman, J. W. Tukey (1974). A projection pursuit algorithm for exploratory data analysis. *IEEE Trans. Comput.*, 23(9), 881–890.

54. Y. Fua, M. Ward, E. Rundensteiner (1999). Hierarchical parallel coordinates for exploration of large datasets. In: D. Ebert, M. Gross, B. Hamann (Eds.), *Proceedings of Visualization '99: Celebrating Ten Years*. IEEE Computer Society Press, Los Alamitos, CA, USA, pp. 43–50.

55. K. Fukunaga (1990). *Introduction to Statistical Pattern Recognition*, 2nd ed. Academic Press.

56. P. Green, F. Carmone, S. Smith (1989). *Multidimensional Scaling: Concepts and Applications*. Allyn and Bacon, Boston.

57. G. G. Grinstein, P. E. Hoffman, R. M. Picket (2002). Benchmark development for the evaluation of visualization for data mining. In: U. Fayyad, G. G. Grinstein, A. Wierse (Eds.), *Information Visualization in Data Mining and Knowledge Discovery*. Morgan Kaufmann Publishers, San Francisco.

58. G. Grinstein, M. Trutschl, U. Cvek (2001). High-dimensional visualizations. In: *Proceedings of Workshop on Visual Data Mining, ACM Conference on Knowledge Discovery and Data Mining*. ACM Press, New York, pp. 1–14.

59. G. G. Grinstein, M. O. Ward (2002). Introduction to data visualization. In: U. Fayyad, G. G. Grinstein, A. Wierse (Eds.), *Information Visualization in Data Mining and Knowledge Discovery*. Morgan Kaufmann Publishers.

60. P. J. F. Groenen (1993). *The Majorization Approach to Multidimensional Scaling: Some Problems and Extensions*. DSWO Press.

61. P. J. F. Groenen, W. J. Heiser (1996). The tunneling method for global optimization in multidimensional scaling. *Psychometrika*, 61, 529–550.

62. P. J. F. Groenen, W. J. Heiser, J. J. Meulman (1998). City-block scaling: smoothing strategies for avoiding local minima. In: I. Balderjahn, R. Mathar, M. Schader (Eds.), *Classification, Data Analysis, and Data Highways*. Springer, pp. 46–53.

63. P. J. F. Groenen, W. J. Heiser, J. J. Meulman (1999). Global optimization in least-squares multidimensional scaling by distance smoothing. *Journal of Classification*, 16(2), 225–254.

64. P. J. F. Groenen, R. Mathar, W. J. Heiser (1995). The majorization approach to multidimensional scaling for Minkowski distances. *Journal of Classification*, 12(1), 3–19.

65. P. Groenen, R. Mathar, J. Trejos (2000). Global optimization methods for multidimensional scaling applied to mobile communication. In: W. Gaul, O. Opitz, M. Schander (Eds.), *Data Analysis: Scientific Modeling and Practical Applications*. Springer, pp. 459–475.

66. J. Han, M. Kamber (2006). *Data Mining, Concepts and Techniques*. Elsevier.

67. H. H. Harman (1976). *Modern Factor Analysis*, 3rd edition. University of Chicago Press, Chicago.

68. M. H. Hassoun (1995). *Fundamentals of Artificial Neural Networks, A Bradford Book*. MIT Press, Cambridge, Masachusetts.

69. T. Hastie (1984). *Principal Curves and Surfaces*. PhD Dissertation, Stanford Linear Accelerator Center, Stanford University, Stanford, California.

70. D. M. Haykin (1999). *Neural Networks: A Comprehensive Foundation*, 2nd ed. Prentice Hall, Upper Saddle River, New Jersey.

71. P. Hellemaa (1998). *The Development of Coastal Dunes and Their Vegetation in Finland*. Dissertation, Fenia 176: 1, Helsinki.

72. G. E. Hinton, R. R. Salakhutdinov (2006). Reducing the dimensionality of data with neural networks. *Science*, 313, 504 – 507.

73. P. E. Hoffman, G. G. Grinstein (2002). A survey of visualizations for high-dimensional data mining. In: U. Fayyad, G. G. Grinstein, A. Wierse (Eds.), *Information Visualization in Data Mining and Knowledge Discovery*. Morgan Kaufmann Publishers, San Francisco.

74. P. E. Hoffman, G. G. Grinstein, D. Pinkney (1999). Dimensional anchors: a graphic primitive for multidimensional multivariate information visualizations. In: *Workshop on New Paradigms in Information Visualization and Manipulation (NPIV'99); November 6, 1999*. ACM press, New York, USA, pp. 9–16.

75. L. Hubert, P. Arabie, M. Hesson-Mcinnis (1992). Multidimensional scaling in the city-block metric: a combinatorial approach. *Journal of Classification*, 9(2), 211–236.

76. J. Hwa, R. M. Graham, D. M. Perez (1995). Identification of critical determinants of α_1-adrenergic receptor subtype selective agonist binding. *Journal of Biological Chemistry*, 270(39), 23189–23195.

77. A. Inselberg (1981). *N-dimensional graphics, part I – lines and hyperplanes*. Technical report G320-2711, IBM Los Angeles Scientific

Center, IBM Scientific Center, 9045 Lincoln Boulevard, LosAngeles (CA), 900435.

78. S. Ivanikovas, V. Medvedev, G. Dzemyda (2007). Parallel realizations of the SAMANN algorithm. In: B. Beliczynski, A. Dzielinski, M. Iwanowski, B. Ribeiro (Eds.), *Adaptive and Natural Computing Algorithms*. *Lecture Notes in Computer Science*, Vol. 4432. Springer, pp. 179–188.

79. A. K. Jain, J. Mao, K. Mohiuddin (1996). Artificial neural networks: a tutorial. *IEEE Computer*, 29(3), 31–44.

80. F. Jara (1998). *A Framework for Extending the Applicability of the Iconographic Technique*. PhD thesis, Institute for Visualization and Perception Research, University of Massachusetts at Lowell.

81. R. Karbauskaitė, O. Kurasova, G. Dzemyda (2007). Selection of the number of neighbours of each data point for the locally linear embedding algorithm. *Information technology and control*, 36(4), 359–364.

82. S. Kaski (1997). *Data Exploration Using Self-Organizing Maps*. PhD thesis, Helsinki University of Technology, Department of Computer Science and Engineering.

83. A. J. Kearsley, R. A. Tapia, M. W. Trosset (1998). The solution of the metric STRESS and SSTRESS problems in multidimensional scaling using Newton's method. *Computational Statistics*, 13, 369–396.

84. D. A. Keim, H.-P. Kriegel (1996). Visualization techniques for mining large databases: a comparison. *IEEE Transactions on Knowledge and Data Engineering*, 8(6), 923–938.

85. D. A. Keim, M. Ward (2003). Visualization. In: M. Berthold, D. J. Hand (Eds.), *Intelligent Data Analysis: an Introduction*. Springer-Verlag, pp. 403–427.

86. H. Klock, J. M. Buhmann (2000). Data visualization by multidimensional scaling: a deterministic annealing approach. *Pattern Recognition*, 33, 651–669.

87. T. Kohonen (2001). *Self-Organizing Maps*, 3rd ed. *Springer Series in Information Science*, Vol. 30. Springer-Verlag.

88. T. Kohonen (2002). Self-Organizing Neural networks: Recent Advances and Applications. In: U. Seiffert, L. C. Jain (Eds.), *Self-Organizing Neural Networks, Studies in Fuzziness and Soft Computing*, Vol. 78. Physica-Verl., Heidelberg, New York, pp. 1–11.

89. A. Konig (2000). Interactive visualization and analysis of hierarchical neural projections for data mining. *IEEE Transactions on Neural Networks*, 11(3).

90. M. A. Kraaijveld, J. Mao, A. K. Jain (1995). A nonlinear projection method based on Kohonen's topology preserving maps. *IEEE Transactions on Neural Networks*, 6(3), 548–559.

91. M. A. Kramer (1991). Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal*, 37(2), 233–243.

92. M. Kraus, T. Ertl (2001). Interactive data exploration with customized glyphs. In: V. Skala (Ed.) *Proceedings of WSCG '01*, pp. 20–23.

93. T. Krilavičius, A. Žilinskas (2008). On structural analysis of parliamentarian voting data. *Informatica*, 19.

94. J. B. Kruskal (1972). Linear transformations of multivariate data to reveal clustering. In: R. N. Shephard, A. K. Romney, S. K. Nerlove (Eds.), *Multidimensional Scaling: Theory and Application in the Behavioural Sciences, I, Theory*. Seminar Press, New York and London.

95. J. B. Kruskal, M. Wish (1978). *Multidimensional Scaling*. Bell Laboratories.

96. O. Kurasova (2005). *Daugiamačių duomenų vizuali analizė taikant savireguliuojančius neuroninius tinklus (SOM)*. Daktaro disertacija, Matematikos ir informatikos institutas, Vilnius.

97. B. Kvedaras, M. Sapagovas (1974). *Skaičiavimo metodai*. Mintis.

98. J. A. Lee, A. Lendasse, N. Donckers, M. Verleysen (2000). A robust nonlinear projection method. In: M. Verleysen (Ed.), *Eighth European Symposium on Artificial Neural Networks, ESANN'2000, Bruges, Belgium*. D-Facto Publications, pp. 13–20.

99. J. A. Lee, A. Lendasse, M. Verleysen (2004). Nonlinear projection with curvilinear distances: isomap versus curvilinear distance analysis. *Neurocomputing*, 57, 49–76.

100. R. C. T. Lee, J. R. Slagle, H. Blum (1977). A triangulation method for the sequential mapping of points from *n*-space to two-space. *IEEE Transactions on Computers*, 26(3), 288–92.

101. H. Levkowitz (1991). Color icons: merging color and texture perception for integrated visualization of multiple parameters. In: G. M. Nielson, L. Rosenblum (Eds.), *Proceedings of IEEE Visualization '91, San Diego, California*, pp. 164–170.

102. H. Lohninger (1994). INSPECT, a program system to visualize and interpret chemical data. *Chemomet. Intell. Lab. Syst.*, 22, 147–153.

103. D. Lowe (1995). Radial basis function networks. In: M. A. Arbib (Ed.), *The Handbook of Brain Theory and Neural Networks*. MIT Press, Cambridge, MA, pp. 779–782.

104. D. Lowe, M.E. Tipping (1996). Feed-forward neural networks and topographic mappings for exploratory data analysis. *Neural Computing and Applications*, 4, 83–95.

105. O. L. Mangasarian, W. H. Wolberg (1990). Cancer diagnosis via linear programming. *SIAM News*, 23(5), 1–18.

106. J. Mao, A. K. Jain (1995). Artificial neural networks for feature extraction and multivariate data projection. *IEEE Transactions on Neural Networks*, 6(2), 296–317.

107. R. Mathar (1997). A hybrid global optimization algorithm for multidimensional scaling. In: R. Klar, O. Opitz (Eds.), *Classification and Knowledge Organization*. Springer, pp. 63–71.

108. R. Mathar (1997). *Multidimensionale Skalierung, Mathematische Grundlagen und Algorithmische Konzepte*. Teubner Verlag.

109. R. Mathar, A. Žilinskas (1993). On global optimization in two-dimensional scaling. *Acta Applicandae Mathematicae*, 33, 109–118.

110. W. S. McCulloch, W. Pitts (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5, 115–133.

111. V. Medvedev (2007). *Tiesioginio sklidimo neuroninių tinklų taikymo daugiamačiams duomenims vizualizuoti tyrimai*. Daktaro disertacija, Vilniaus Gedimino technikos universitetas, Technika, Vilnius.

112. V. Medvedev, G. Dzemyda (2006). Optimization of the SAMANN network training. *Journal of Global Optimization*, 35(4), 607–623.

113. Z. Michalewicz (1996). *Genetic Algorithms + Data Structures = Evolution Programs*. Springer, Berlin.

114. R. S. Michalski (1978). *A Planar Geometric Model for Representing Multidimensional Discrete Spaces and Multiple-Valued Logic Functions*. Technical Report UIUCDCSR-78-897, University of Illinois at Urbaba-Champaign.

115. H. Miyano, Y. Inukai (1982). Sequential estimation in multidimensional scaling. *Psychometrika*, 47, 321–336.

116. G. P. Nason (1992). *Design and Choice of Projection Indices*. Ph.D. thesis, University of Bath, Bath, U.K.

117. T. R. Nelson, J. Rabianski (1988). Consumer preferences in housing market analysis: an application of multidimensional scaling techniques. *Real Estate Economics*, 16, 138–159.

118. E. Oja (1991). Data compression, feature extraction, and autoassociation in feedforward neural networks. *Artificial Neural Networks*, 287(1), 737–745.

119. O. Opitz, A. Hilbert (2000). Visualization of multivariate data by scaling and property fitting. In: W. Gaul, O. Opitz, M. Schader (Eds.), *Data Analysis: Scientific Modeling and Practical Applications*. Springer, pp. 505–514.

120. R. M. Pickett, G. Grinstein (1988). Iconographic displays for visualizing multidimensional data. In: *IEEE Conference on Systems, Man, and Cybernetics*, pp. 514–519.

121. S. M. Pincus (1991). Approximate entropy as a measure of system complexity. *Procedings of National Academy of Sciences of the United States of America*, 88(6), 2297–2301.

122. D. A. Rabenhorst (1994). Interactive exploration of multidimensional data. In: *Proceedings of the SPIE Symposium on Electronic Imaging*, Vol. 2179, pp. 277–286.

123. Š. Raudys (2001). *Statistical and Neural Classifiers: an Integrated Approach to Design*. *Advances in Pattern Recognition*. Springer-Verlag.

124. W. Ribarsky, E. Ayers, J. Eble, S. Mukherjea (1994). Glyphmaker: creating customized visualization of complex data. *IEEE Computer*, 27(7), 57–64.

125. S. T. Roweis, L. K., Saul (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500), 2323–2326.

126. J. O. Ruuskanen, J. Laurila, H. Xhaard, V.-V. Rantanen, K. Vuoriluoto, S. Wurster, A. Marjamaki, M. Vainio, M. S. Johnson, M. Scheinin (2005). Conserved structural, pharmacological and functional properties among the three human and five zebrafish α_2-adrenoceptors. *British Journal of Pharmacology*, 144(2), 165–177.

127. A. Sachinopoulou (2001). *Multidimensional Visualization*. Julkaisuvuosi.

128. J. W. Sammon (1969). A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, 18, 401–409.

129. T. D. Sanger (1989). Optimal unsupervised learning in a single-layer linear feedforward neural network. *Neural Networks*, 2, 459–473.

130. S. S. Schiffman, M. L. Reynolds, F. W. Young (1981). *Introduction to Multidimensional Scaling: Theory, Methods, and Applications*. Academic Press.

131. R. Silipo (2003). Neural networks. In: M. Berthold, D. J. Hand (Eds.), *Intelligent Data Analysis: an Introduction*. Springer-Verlag, pp. 269–320.

132. V. Šaltenis, J. Aušraitė (2002). Data Visualization: ideas, methods, and problems. *Informatics in Education*, 1, 129–148.

133. Y. Takane (2006). Applications of multidimensional scaling in psychometrics. *Handbook of Statistics*, 26, 359–400.

134. M. Takatsuka (2001). An application of the self-organizing map and interactive 3-D visualization to geospatial data. In: *Proceedings of the 6th International Conference on GeoComputation*. Brisbane, Australia, CD-ROM.

135. P. Taylor (2003). Statistical methods. In: M. Berthold, D. J. Hand (Eds.), *Intelligent Data Analysis: an Introduction*. Springer-Verlag, pp. 69–129.

136. S. Telser, M. Staudacher, Y. Ploner, A. Amann, H. Hinterhuber, M. Ritsch-Marte (2004). Can one detect sleep stage transitions for on-line sleep scoring by monitoring the heart rate variability. *Somnology*, 8(2), 33–41.

137. J. B. Tenenbaum, V. de Silva, J. C. Langford (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500), 2319–2323.

138. M. E. Tipping (1996). *Topographic Mappings And Feed-Forward Neural Networks*. Dissertation, Aston University, Birmingham.

139. W. S. Torgerson (1958). *Theory and Methods of Scaling*. John Wiley and Sons.

140. S. Uhlén, M. Dambrova, J. Näsman, H. B. Schiöth, Y. Gu, A. Wikberg-Matsson, J. E. S. Wikberg (1998). [^{3}H]RS79948-197 binding to human, rat, guinea pig and pig α_{2A}-, α_{2B}- and α_{2C}-adrenoceptors. Comparison with MK912, RX821002, rauwolscine and yohimbine. *European Journal of Pharmacology*, 343(1), 93–101.

141. A. Ultsch, H. P. Siemon (1990). Kohonen's self-organizing feature maps for exploratory data analysis. In: *Proceedings of International Neural Network Confererence (INNC'90), Dordrecht, Netherlands*. Kluwer Academic Publishers, pp. 305–308.

142. M. C. Van Wezel, W. A. Kosters (2004). Nonmetric multidimensional scaling: neural networks versus traditional techniques. *Intelligent Data Analysis*, 8(6), 601–613.

143. A. Varoneckas, A. Žilinskas, J. Žilinskas (2006). Multidimensional scaling using parallel genetic algorithm. In: I. D. L. Bogle, J. Žilinskas (Eds.), *Computer Aided Methods in Optimal Design and Operations*. *Series on Computers and Operations Research*, Vol. 7. World Scientific, pp. 129–138.

144. A Verikas, A. Gelžinis (2003). *Neuroniniai tinklai ir neuroniniai skaičiavimai*. Technologija, Kaunas.

145. M. Ward (1994). XmdvTool: integrating multiple methods for visualizing multivariate data. In: *IEEE Visualization'94*. IEEE Computer Society Press, Los Alamitos, CA, USA, pp. 326–333.

146. M. Ward, J. LeBlanc, R. Tipnis (1994). *N*-land: a graphical tool for exploring *n*-dimensional data. In: *CG194 Proc: Insight Through Computer Graphics*, pp. 130–141.

147. R. Wegenkittl, H. Loffelmann, E. Grller (1997). Visualizing the behavior of higher dimensional dynamical systems. In: *Proceedings of IEEE Visualization 1997*. IEEE Computer Society Press, Los Alamitos, CA, USA, pp. 119–125.

148. E. J. Wegman, Q. Luo (1996). *High Dimensional Clustering Using Parallel Coordinates and the Grand Tour*. Technical report No. 124, Center for Computational Statistics, George Mason University.

149. C. M. Wittenbrink, A. T. Pang, S. K. Lodha (1996). Glyphs for visualizing uncertainty in vector fields. *IEEE Transactions on Visualization and Computer Graphics*, 2(3), 266–279.

150. P. C. Wong, R. D. Bergeron (1997). 30 years of multidimensional multivariate visualization. In: G. M. Nielson, H. Hagan, H. Muller (Eds.), *Scientific Visualization – Overviews, Methodologies and Techniques*. IEEE Computer Society Press, Los Alamitos, CA, pp. 3–33.

151. L. Yang (2004). Sammon's nonlinear mapping using geodesic distances. In: *ICPR'04: Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04)*, Vol. 2. IEEE Computer Society, Washington, DC, pp. 303–306.

152. K. Y. Yeung, W. L. Ruzzo (2001). *An Empirical Study on Principal Component Analysis for Clustering Gene Expression Data*. Technical Report UW-CSE-01-04-02, University of Washington.

153. M. Žičkus (1999). *Meteorologinių parametrų įtaka miesto oro užterštumui ir jo prognozavimas*. Daktaro disertacija, Vilniaus universitetas.

154. A. Žilinskas, J. Žilinskas (2006). On multidimensional scaling with Euclidean and city block metrics. *Technological and Economic Development of Economy*, 12(1), 69–75.

155. A. Žilinskas, J. Žilinskas (2006). On visualization of multidimensional data using three-dimensional embedding space. *Technological and Economic Development of Economy*, 12(4), 353–359.

156. A. Žilinskas, J. Žilinskas (2006). Parallel hybrid algorithm for global optimization of problems occurring in MDS-based visualization. *Computers and Mathematics with Applications*, 52(1–2), 211–224.

157. A. Žilinskas, J. Žilinskas (2007). Two level minimization in multidimensional scaling. *Journal of Global Optimization*, 38(4), 581–596.

158. A. Žilinskas, J. Žilinskas (2008). A hybrid method for multidimensional scaling using city-block distances. *Mathematical Methods of Operations Research*.

159. J. Žilinskas (2006). Multidimensional scaling in protein and pharmacological sciences. In: I. D. L. Bogle, J. Žilinskas (Eds.), *Computer Aided Methods in Optimal Design and Operations*. *Series on Computers and Operations Research*, Vol. 7. World Scientific, Singapore, pp. 139–148.

160. J. Žilinskas (2007). Reducing of search space of multidimensional scaling problems with data exposing symmetries. *Information Technology and Control*, 36(4), 377–382.

Gintautas DZEMYDA, Olga KURASOVA, Julius ŽILINSKAS
DAUGIAMAČIŲ DUOMENŲ VIZUALIZAVIMO METODAI

Vadovėlis informatikos krypties doktorantams ir magistrantams

Redaktorė Zita Manstavičienė

Viršelio dailininkė Giedrė Stapčinskienė

Išleido Matematikos ir informatikos institutas
Akademijos g. 4, LT-08663 Vilnius
Tel. (8 5) 2109300, faks. (8 5) 2729209
El. paštas: mathematica@ktl.mii.lt

Spausdino UAB „Mokslo aidai“
Goštauto g. 12, LT-01108 Vilnius
Tiražas 300 egz.
Užsakymo Nr. 1705